REVISTA PARA USUARIOS DE

Drean commodore

Nº 2 ₳ 2.00

REP. ARGENTINA

Vacaciones con la computadora

Procesador de texto

EL ARTE DE EDITAR

Primeros pasos

MICROPROCESADOR

SIMONS' BASIC

CODIGO DE MAQUINA

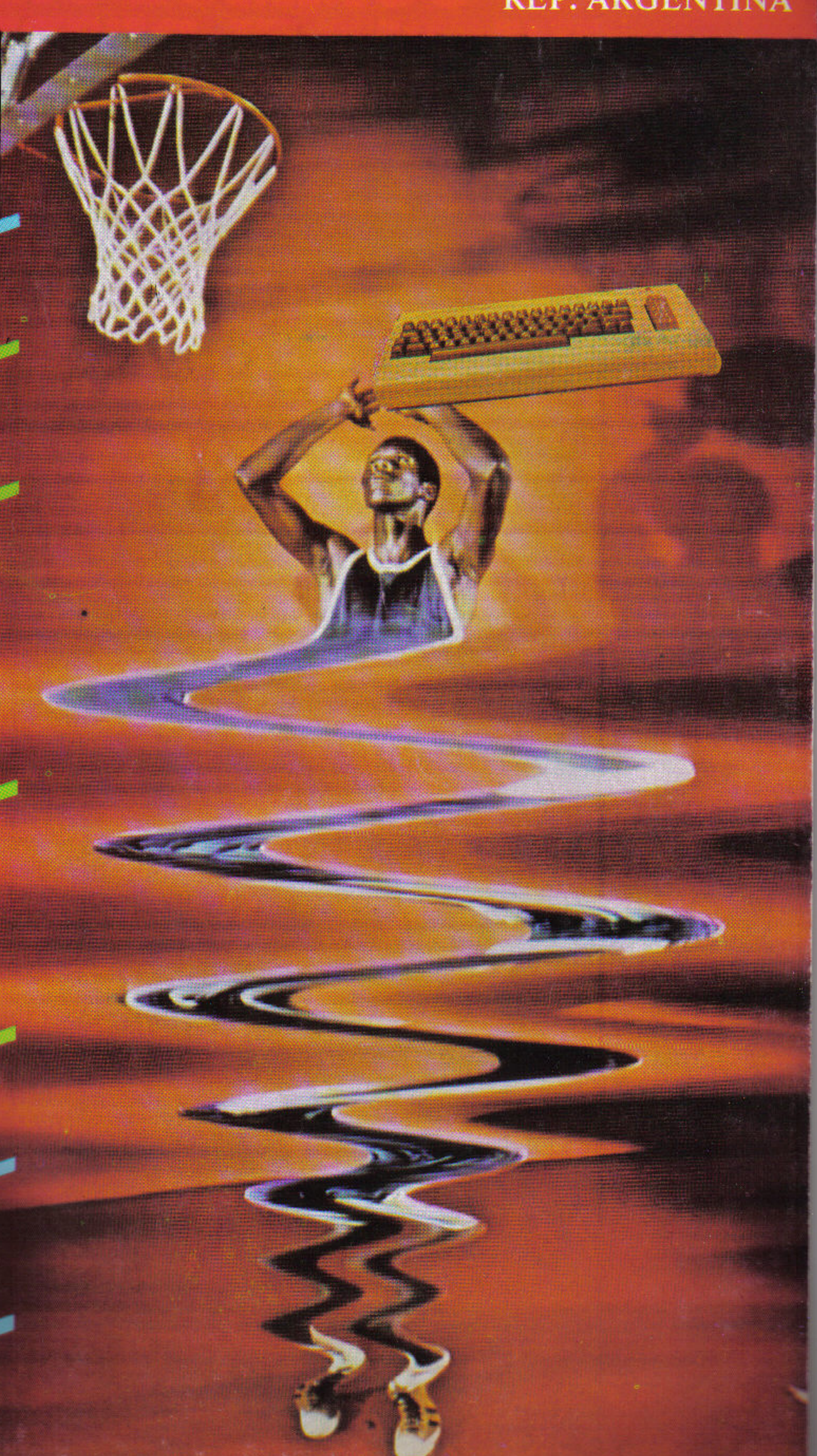
Programas

- SNAKER
- ASSEMBLER 1.1

Drean Commodore 16

INSTRUCCIONES PROPIAS

Software para jugar



SUMARIO

NOTAS TECNICAS

PROGRAMAS

REVISION DE SOFTWARE

SECCIONES FIJAS

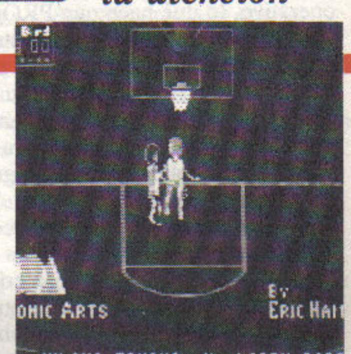


Explicamos los comandos que constituyen este Basic Plus

HELP
F7

Las teclas que más llaman la atención





Continuamos comentando los programas que Peek presentó en el mercado argentino para la Drean Commodore 64

Drean Commodore AÑO 1 - Nº 2

Director General
Ernesto del Castillo
Director Editorial
Cristian Pusso
Director Periodistico
Fernando Flores
Director Financiero
Javier Campos Malbrán

Asesor Editorial
Armengol Torres Sabaté
Arte y Diagramación
Fernando Amengual
Coordinador
Ariel Testori
Secretaria
Moni Ocampo

Departamento de Avisos
Oscar Devoto
Departamento de Publicidad
Jefe: Dolores Urien
Promotora: Marita García

Drean Commodore es una Revista mensual editada por Editorial PROEDI S.A. (e./f.), Cerrito 1320, 1º Piso, Buenos Aires, Te.: 42-9681/9. Registro Nacional de la Propiedad Intelectual: E.T. M. registrada.

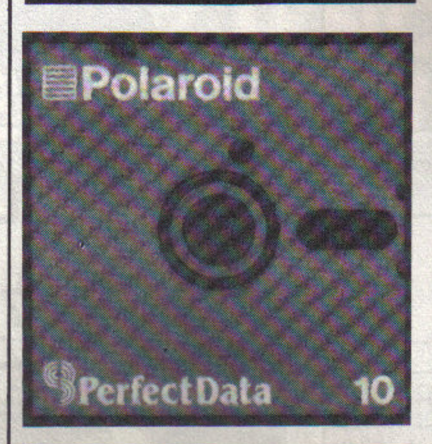
Precio de este ejemplar: ₳ 2.
Impresión: Calcotam. Fotocromo tapa: Columbia. Fotocomposición: Van Waveren.
Los ejemplares atrasados se venderán al precio del último número en circulación.

Nuevo lenguaje.

La firma Systems Management lanzó al mercado norteamericano un nuevo producto denominado PROMAL. Es un nuevo lenguaje de alto nivel para la Commodore 64.
PROMAL significa Programmers Micro Aplication Language. De por sí, es un lenguaje compilable y su estructura es muy similar al de PASCAL o al de "C". Se distinguen tres secciones en las que se pueden definir constantes y variables globales, procedimientos y funciones, y por último, el programa o módulo principal. PROMAL está orientado hacia la programación estructurada. Posee sentencias tales como el WHILE, REPEAT, UNTIL, PROC, BEGIN, FUNC, ELSE, etc. Al lenguaje lo acompañan un editor y el sistema operativo, lo que hace que PROMAL sea un sistema complejo. Al sistema operativo también se lo conoce con el nombre de "EJECUTIVO". Su función es suministrar al programador las herramientas necesarias para que pueda escribir y probar sus programas (por supuesto en PROMAL).
Es decir que el EJECUTIVO acepta y ejecuta los comandos ingresados desde el teclado. De acuerdo con la crítica calificada, PROMAL es un lenguaje muy potente y fácil de usar. Además es lo más notable en productos recientemente lanzados para Commodore.

Forth para la C-64

Forth es un lenguaje de alto nivel para computadoras de cuarta generación. Es un lenguaje intérprete como el Basic, debido a ello los requerimientos de memoria para escribir los programas son mínimos (aún más pequeños que en Basic). Básicamente FORTH se compone de cinco módulos:
INDICE: Contiene todas las instrucciones del lenguaje.
ASSEMBLER: Nos permite trabajar con el assembler de la C-64.
STACK: Ejecuta todas las operaciones aritméticas.
MEMORIA: Aquí se realizan todas las operaciones internas del FORTH. También se la conoce con el nombre de memoria de trabajo.
INTERPRETE: Toma el siguiente texto Forth, lo decodifica a código de máquina y lo ejecuta (igual que en Basic).
Con respecto al STACK podemos decir que su modo de operación es LIFO (Last in First Out: último en entrar primero en salir). Es decir que la notación matemática que se utiliza es la Polaca Inversa. Por ejemplo si deseamos efectuar la operación 2+3 esto es en FORTH 2 3 +.
El lenguaje está disponible en diskette o en cartridge. Y lo acompaña un extenso y explicativo manual.



Nuevos desarrollos.

La compañía POLAROID acaba de lanzar nuevos diskettes de 5 1/4 pulgadas, de alta densidad y doble lado. La capacidad de éstos es de 1,6 Mbyte, es decir 1.600.000 bytes. Estos diskettes pueden ser cargados en el Disk Drive 1541.

ADSOR: Consultor Lógico

ADSOR es un sistema desarrollado en base a Inteligencia Artificial. Nos permite escribir programas que son automáticamente compilados en este lenguaje PENSANTE el cual habilita a la computadora para replicar razonamientos lógicos y ayudarnos en la resolución de todo tipo de problemas.

Sistema para aplicaciones matemáticas

Este programa, desarrollado para la C-64, realiza todo tipo de operaciones matemáticas. Su gama de funciones va desde simples sumas hasta complejas interpolaciones. Alguna de sus ventajas es la posibilidad de calcular la dimensión de cualquier figura geométrica que dibujemos sobre la pantalla (por supuesto en alta resolución).

DATAMAT-64

DATAMAT-64 es una poderosa base de datos desarrollada para la C-64 juntamente con la unidad de disco 1541. DATAMAT-64 nos permite definir más de 50 campos por registro pudiendo almacenar 2000 registros por diskette. También permite clasificar la base acorde al campo seleccionado como así también imprimir determinados registros con el formato designado por nosotros. Fue desarrollado por Abacus Software y está disponible en diskette (lo acompaña su correspondiente manual).

Pascal para la C-64 y C-128

Abacus Software ha desarrollado un completo sistema correspondiente al lenguaje PASCAL para la Commodore 64 y 128 denominado SUPER PASCAL. Incluye un extenso compilador, un editor de archivos, un assembler integrado y un paquete utilitario para la administración de archivos y del disco. Acompañan al sistema el manual correspondiente con más de 200 páginas.

Amiga

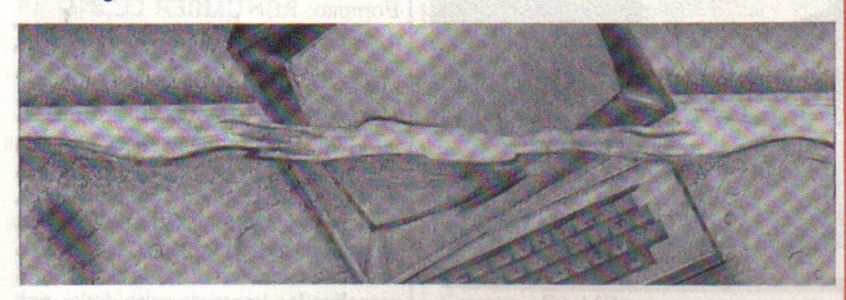
Acaba de salir en el mercado norteamericano la nueva PC de Commodore Computers denominada "AMIGA". La crítica la define como una de las más sofisticadas PC actuales, que compite con las tradicionales de APPLE o IBM. Una de las tantas características de "AMIGA" es su microprocesador 68000 (Motorola). Este micro de 16/32 bits aumenta la velocidad de operación notablemente y, además, puede direccionar 8 Megabytes de memoria.
Para la parte de graficación "AMIGA" es ideal. Permite trabajar con 4096 colores distintos (por supuesto en alta resolución). El texto se edita en 40 u 80 columnas. Junto a ella viene el tradicional "Mouse", el cual nos permite trabajar con textos, menús, etc. Drean tiene la intención de producirla en la Argentina, como anticipamos en el número anterior.

Lenguaje ADA

ADA es un lenguaje de alto nivel. El nombre no representa siglas como, por ejemplo, las de BASIC (Beginners All-purpose Symbolic Instruction Code), no hace alusión a las hadas, sino que es el nombre de una mujer que allá por el siglo XVIII se dedicaba a desarrollar nuevas estructuras lógicas. Fomenta la programación modular-estructurada siendo, además, muy fácil de aprender.

VACACIONES CON LA COMPUTADORA

A muchos les gustaría incluir la Commodore en su equipaje, pero para no encontrarse con sorpresas desagradables ofrecemos algunos consejos útiles.



Protección del equipo

Siempre que transportemos la computadora asegurémonos de que esté bien empacada y protegida. Existen diferentes estuches para equipos de computación que brindan la seguridad buscada. Pero una forma menos cara de empacar el equipo es utilizando los estuches originales. Estas cajas y los pedazos de espuma de plástico con que vienen, proveen una excelente protección.
Si las tiramos igualmente podemos reemplazarlos por gomaespuma o algodón. Se ha descubierto que las toallas, medias y remeras, además del uso usual también son excelentes como "paragolpes". Si de todas maneras deberemos cargar con todo eso, ¿por qué no tratar de usarlos como protección?
Los seguros de viaje son una buena forma para protegerse de pérdidas monetarias en caso de accidentes. Generalmente son baratos y, usualmente, pueden obtenerse donde se compran los cheques de viajero. Algunas tarjetas de crédito importantes también incluyen este beneficio.
Y a veces las pólizas de seguros de propiedades o de alquiler preveen seguros para viajes o los hacen factibles.

Cuando el viaje es fuera del país

Si la intención es viajar al exterior y llevar la Commodore, hay varias cosas que se deben tener en cuenta. Primero y principal: no toda la electricidad se genera de la misma manera. En nuestro país, la intensidad es de 210 a 220 voltios. Mientras que en los Estados Unidos y naciones de Europa el voltaje es diferente. Sin ser un técnico, se puede decir que el equipo no trabajará con electricidad "extranjera" si no tiene algo para hacer la conversión. Por lo tanto habrá que llevar un adaptador de voltaje.
Siguiendo en el tema, recordamos que la Drean Commodore no se adaptará tampoco a un televisor que tenga diferente sistema. En Argentina, la televisión funciona bajo la norma PAL-N y en otros países se usa NTSC o SECAM. Llevarse un televisor o un monitor portátil propio puede ser una buena idea. Aunque no se puedan ver los programas locales de televisión, sí disfrutaremos los videos de Drean Commodore.
Cuando el viaje sea fuera de la Argentina, recordemos que tendremos que pasar por la aduana. No olvidemos declarar la computadora cuando entremos al país a visitar o tendremos problemas al abandonar el lugar. En la aduana proveerán de una estampilla para el equipo o nos darán un pase. No olvidemos que no hay nada más molesto para empañar un viaje de vacaciones que tener problemas de aduana. Para estas situaciones es útil llevar fotocopia de los recibos de compra del equipo.

Viajando por avión

Sean donde sean las vacaciones, si viajamos por avión deberemos atravesar las tradicionales medidas de seguridad de todo aeropuerto. Estas incluyen examinar el equipaje a través de rayos X y pasar por el detector de metales. A pesar de que en los mismos aeropuertos se informa que los rayos X son inofensivos para los equipos, es aconsejable no correr el riesgo.
Para el traslado del equipo es importante recordar que para evitar cualquier tipo de interferencia magnética en los diskettes o cassettes, lo mejor es llevarlos en un bolso de mano y embalar cuidadosamente el resto del equipo que se despachará normalmente.

Evitemos desastres

Independientemente del lugar que se haya elegido para pasar las vacaciones es importante no dejar de observar algunas reglas generales.
Nunca usemos la computadora cerca de una piscina, río, mar, etc., es decir cerca del agua. Además del daño que la Commodore o sus periféricos podrán sufrir, nos exponemos a una eventual descarga eléctrica fatal.
El mismo Sol que nos dejará como un camarón, arruinará el equipo que es sutilmentte susceptible a los daños producidos por un calor excesivo. No sólo se derretirán las partes plásticas sino que además se perjudicarán los componentes electrónicos internos. La computadora no necesita un bronceador más intenso del que ya posee. ¡No la cocinemos!
La arena y el oleaje son espléndidos para los humanos pero las computadoras deben huír de ellos como de una plaga. La arena podría filtrarse en el teclado, en el disk-drive o en el impresora de la computadora y provocar un verdadero desastre. Un solo granito de arena haría un estrago en la superficie magnética del diskette o del cassette.
El aire salado del mar corroerá el metal y los componentes electrónicos de la computadora. El segundo enemigo es la humedad que acarrea el aire marino. La sal y la humedad son dos duendes malignos que provocarán el mal funcionamiento de la computadora.

En Estados Unidos

Los campamentos, cruceros, hoteles y otros lugares en donde puedan combinarse turismo y cursos de programación y computación se están haciendo cada vez más populares en los Estados Unidos. Los precios, por supuesto, varían enormemente de acuerdo al lugar que uno elija y al tipo de comodidades que se ofrecen.
Donde sea que pasen vacaciones, en el país o recorriendo el mundo, esperamos que lo pasen muy bien.

SIMONS' BASIC

Está disponible en cartridge, diskette y cassettes, y demuestra la versatilidad de la C-64.



"Cuando David Simons cumplió 13 años, su padre le regaló una Commodore 64. Desde ese momento él fue estudiando y entendiendo el funcionamiento de la computadora. Luego de tres años, insertó nuevos comandos y sentencias a la C-64 constituyendo de esta manera lo que hoy se conoce con el nombre de SIMONS "BASIC".

Estas palabras corresponden a una parte de "datos del autor" perteneciente al manual de este nuevo Basic. Como habrán notado, su autor fue un joven estudiante de 16 años.

El objetivo de esta nota es comentar y explicar los comandos que constituyen este Basic Plus. Con ellos y los standard, el programador dispone de 114 comandos orientados a consistencia de entrada y manipulación de texto, gráficos, manipulación de pantalla, Sprites y gráficos definidos por el usuario, tratamiento de errores, programación estructurada, efectos de sonido, lectura de funciones, manejo del Disk Drive, etc. Estos fueron escritos totalmente en lenguaje de máquina con lo que se demuestra lo versátil del sistema operativo de la C-64 (no todas las home computers permiten una ampliación de su Basic).

SIMONS' BASIC está disponible en Cartridge, Diskette o Cassette. Ocupa un total de 8 K lo que implica que 30 K están disponibles para el programador. Luego de cargarlo y activarlo, aparecerá en pantalla el siguiente mensaje:

EXPANDED CBM V2 BASIC
30719 BYTES FREE

A partir de aquí, estamos en condiciones de trabajar con cualquier nueva sentencia o comando. Por lo pronto, empezaremos explicando el:

Comando: KEY
Formato: KEY N,"COM"
donde N es un número comprendido entre 1 y 16

COM: es un comando válido del Simons
Función: Asigna el comando especificado por COM a la tecla de función especificada por N.
A través de ésta podemos asignar a las teclas de función (f1, f2,..., f8,... f16) los comandos que con más frecuencia son utilizados por nosotros durante la realización de un programa. De esta manera nos basta con presionar una de ellas para que un comando determinado se ejecute o, simplemente, sea impreso en pantalla.
Ejemplo: Deseamos asignar el comando LIST a f1 y el RUN a f8.

```
KEY1,"LIST"+CHR$(13) (cr)
KEY8,"RUN"+CHR$(13) (cr)
```

También pudimos haber hecho:

```
KEY1,"LIST" (cr)
KEY8,"RUN" (cr)
```

La diferencia que hay entre estos dos grupos es que, los del primero, se ejecutan directamente ya que acompañan al comando el código de Return (CHR$(13)). es como tipear LIST y luego RETURN. En el segundo grupo sólo se imprime el comando sin ejecutarse. Si nosotros deseamos esto deberemos presionar RETURN.

NOTA: La Commodore dispone de ocho teclas de función activadas por f1-f3-f5-f7-SHIFT f1-SHIFT f3-SHIFT f5-SHIFT f7; las restantes 8 se activan a través de la tecla Commodore, Alpha Loc, y las teclas de función.

Comando: AUTO
Formato: AUTO LC,INC
donde:
LC: Línea de comienzo
INC: Incremento
Función: Genera automáticamente los números de línea de un programa a partir de la línea representada por LC con incrementos INC.
Con auto nos evitamos tener que tipear cada línea del programa. Para salir del modo sólo debemos presionar RETURN a continuación de la línea actual.

Comando: RENUMBER
Formato: RENUMBER LC,INC
donde:
LC e INC representan los mismos parámetros de comando AUTO.
Función: Reenumera las líneas del programa actual comenzando con LC y a pasos de INC.
Este comando es muy útil cuando necesitamos intercalar nuevas líneas de programa. No obstante, no es lo suficientemente eficaz ya que no actualiza las líneas de salto dadas por GOTO o GOSUB. Es decir que si nuestro programa es:

```
 5 REM PROGRAMA SUMADOR
10 INPUT "INGRESE DOS NUMEROS";A,B
27 PRINT "SU SUMA ES:";A+B
29 GOTO 10
```

Y hacemos:

```
RENUMBER 10,5 (cr)
```

el programa queda:

```
10 REM PROGRAMA SUMADOR
15 INPUT "INGRESE DOS NUMEROS";A,B
20 PRINT "SU SUMA ES:"; A+B
25 GOTO 10
```

Notarán que el GOTO de la línea 25 quedó sin actualizar. Lamentablemente esta actualización la debemos efectuar nosotros.

Comando: PAUSE
formado: PAUSE "MEN",TE
ó
PAUSE TE
donde
MEN: Mensaje
Te: Tiempo de Espera (en segundos)
Función: Suspende la ejecución del programa actual el tiempo representado por TE. Opcionalmente, podemos imprimir un mensaje aclarativo mientras efectuamos este comando.
Ejemplo:

```
10 PAUSE "ESPERE 10 SEGUNDOS ANTES DE CONTINUAR", 10
```

ó

```
10 PAUSE 10
```

Comando: OLD
Formato: OLD
Función: Reestablece el programa luego de haber ejecutado el mando NEW.
Si, por algún motivo, tipeamos el comando NEW o reseteamos la C-64

POTENTE PLUS PARA LA C-64

durante la carga de nuestro programa, podemos recuperarlo utilizando OLD. Un pequeño ejemplo de ello sería:

```
10 REM PRUEBA OLD
20 PRINT "PRUEBA DE OLD"
LIST (cr)
10 REM PRUEBA OLD
READY
NEW (cr)
READY
LIST (cr)
READY
OLD (cr)
READY
LIST (cr)
10 REM PRUEBA OLD
20 PRINT "PRUEBA DE OLD"
READY
```

Comando: FIND
Formato: FIND "COD"
donde
COD: String o grupo de caracteres a buscar
Función: Imprime el número de la/s línea/s donde se encuentra el string buscado.
Este comando nos ayuda a localizar una serie de caracteres dentro de nuestro programa. Si los encuentra, imprimirá los números de las líneas correspondientes (en caso en que esté en más de una), caso contrario no imprime nada. Una de las desventajas que posee es que el string a buscar debe ser, exactamente, igual al que se encuentra en el programa. Es decir que si queremos saber si el string "TEST" se encuentra debemos ingresar "TEST" y no " TEST". En este caso el mando FIND no lo encontrará. Además si este string se halla en el programa, en particular se hallará también todo substring de él como ser " TE", "ST", etc. Esta es otra de las desventajas del comando ya que no puede hallar substrings del string buscado.
Ejemplo:

```
10 REM PRUEBA DE FIND
11 PRINT "COM FIND"
FIND "COM FIND" (cr)
READY
FIND "COM FIND" (cr)
11
READY
```

Comando: DUMP
Formato: DUMP
Función: Imprime las variables y sus contenidos que utiliza el programa actual.
Ejemplo: Si realizamos en modo directo:

```
A=34 : B=2265 : C$= "PRUEBA DE DUMP" (cr)
READY
DUMP (cr)
A=34
B=2265
C$= "PRUEBA DE DUMP"
READY
```

Comando: COLD
Formato: COLD
Función: Resetea la C-64 e inicializa el SIMONS 'BASIC
Este comando borra todo programa residente en la memoria e imprime el encabezamiento del inicio de SIMONS 'BASIC. De todas maneras, el programa destruido puede ser recuperado a través del OLD.
Comando: OPTION
Formato: OPTION N
donde
N: es un número comprendido entre 0 y 255.
Función:
Este comando, con un parámetro de 10, produce que todos los comandos del SIMON sean impresos en campo inverso cuando se solicita un LIST (ya sea en pantalla o sobre impresora). Cualquier otro valor de N (distinto de 10) "apaga" este comando.

EL CODIGO DE MAQUINA

Los programadores lo utilizan cuando requieren velocidad en determinados procedimientos. ¿Cómo generarlos? Esta nota contesta esa pregunta.



Hemos visto, en el número anterior, las características que posee el Assembler juntamente con las diferencias básicas que existen con el lenguaje Basic. Hablamos sobre su velocidad de ejecución, ideal para realizar juegos o controlar procesos externos al computador, tal como verificar la temperatura de calderas tipo Sedroc, contabilizar el tiempo exacto que necesita un proceso químico, etc.
A partir de este número empezaremos a profundizar un poco más, sobre el Assembler de la C-64 y de su código de máquina (aquí lo llamaremos C.M.) En rigor tendríamos que comenzar con el absoluto (C.M.) ya que fue el primero en aparecer y, por ende, en utilizarse. Los programadores, luego de experimentar la complejidad que surgía de desarrollar programas en binario y/o hexadecimal, decidieron realizar un "traductor" cuya entrada era un cierto código preestablecido (mnemotécnicos) y cuya salida era en C.M.

Binario-Hexadecimal

Necesitamos, antes de iniciarnos en esto, conocer un poco sobre esos dos sistemas de numeración. Generalmente se utiliza el sistema decimal cuya base es, obviamente, 10. Todo número se puede representar como una combinación de potencias de 10 juntamente con los elementos que constituyen este "conjunto". Ellos son 0,1,2,3,...,9. Por ejemplo el número 2451 se representa como
$2*10\uparrow3*4*10\uparrow2+5*10\uparrow1+1*10\uparrow0$
(recordemos que todo número, distinto de cero, elevado a la potencia cero es uno). Con los sistemas Binarios y Hexadecimales ocurre lo mismo con la salvedad que sus bases y sus elementos son 2; 0 y 1 para el primero, 16;0,1,2,...,9,A,B,C,D,E,F para el segundo. Para pasar un número cualquiera Binario a decimal realizamos la misma combinación que antes pero, ahora, utilizamos 2 en lugar de 10. Por ejemplo si tenemos el número binario 1101, su correspondiente en decimal es:
$1*2\uparrow3+1*2\uparrow2+1*2\uparrow1+0*2\uparrow0$
Si realizamos esta operación comprobaremos que su resultado es 13 (decimal). Si tenemos el número 1R2F efectuamos la operación similar teniendo en cuenta el valor que poseen las letras. (A=10,B=11,C=12, D=13,E=14,F=15). Acorde a esto tendríamos:
$1*16\uparrow3+10*16\uparrow2+2*16\uparrow1+15*16\uparrow0$
lo que nos daría 6703 (también decimal).
Habrán notado que siempre tomamos el dígito de la derecha y lo multiplicamos por la base del sistema elevada a la cantidad de dígitos que posee el número menos uno. A continuación tomamos el dígito que sigue, lo multiplicamos por la base elevada a uno menos que el anterior. Continuamos así hasta tomar todos los dígitos.
Por último necesitamos saber como se realiza el pasaje inverso es decir de decimal a binario o hexadecimal. El método que se utiliza comúnmente es el de divisiones sucesivas. Dado un número decimal, se lo divide por la base del sistema deseado. Por ejemplo si el número es 1220 y lo que queremos pasar a binario el procedimiento es:

```
122 |2
 02  61 |2
  0  01  30 |2
      1  10  15 |2
          0   1   7 |2
                  1   3 |2
                      1   1 |2
                          1   0
```

122 = 1111010

Si el mismo número, lo pasamos a hexadecimal:

```
122 |16
 10   7 |16
      7   0
```

122 = $7A

Es decir que el número buscado está formado por los restos sucesivos de las divisiones (empezando de abajo hacia arriba).
La comprobación se las dejamos a ustedes. Simplemente utilicen el procedimiento inverso y comprueben que se llega al mismo número. La Tabla 1 les ofrece la representación de los números del 1 al 15 con su correspondientes equivalencias.

Un poco de arquitectura

Todas las instrucciones de un microprocesador involucran a casi todos los registros de trabajo de la C.P.U. Estos son: ACUMULADOR, REGISTRO X, REGISTRO Y, STACK POINTER, REGISTRO DE ESTADO DEL MICROPROCESADOR, CONTADOR DE PROGRAMA. Es necesario conocer internamente la unidad central de proceso a fin de entender y utilizar correctamente las instrucciones en C.M.
¿De qué sirve saber que A2 es cargar el registro X en forma inmediata, si no sabemos qué es este registro?, o que 48 es almacenar el contenido actual del Acumulador en el Stack, sin saber qué es esto.
En este mismo número hay una nota que habla sobre la arquitectura del micro 6510. Antes de continuar con la lectura de esta, les pedimos que lean primero aquélla.

Los primeros pasos

Nuestra primer serie de programas será en C.M. En el futuro, desarrollaremos los programas en C.M. juntamente con su equivalente en mnemotécnico. Para realizar lo primero utilizaremos el "editor de absoluto" del número anterior denominado ASSEMBLER 1.1.
El primer ejercicio consiste en cargar el acumulador en forma inmediata, con, por ejemplo, FF (255 decimal). Si nos fijamos en el set de instrucciones comprobaremos que el código de operación es A9.
Tomando como dirección inicial $3000 (el $ indica que se trata de un número hexadecimal, si no se especifica se trata

de un número decimal):
3000 ? A9
3001 ? FF
3002 ? 60 (finalizamos el programa)
De esta manera acabamos de escribir en tres posiciones de memoria consecutivas el programa deseado. Como hace el micro para saber que tiene que cargar el acumulador con FF y no con 60? La respuesta es que la carga que se realiza es inmediata, es decir que el micro sabe que lo que sigue al código de operación es el operando. Existen otras formas de cargar el acumulador, como por ejemplo en forma absoluta. Todos estos modos de direccionamiento los veremos más adelante.
Lo que debemos hacer ahora es transferir el control del micro para ejecutar dicho programa. Esto lo logramos (en nuestro editor) tipeando R:
3003 ? .
? R
Para comprobar que el contenido del acumulador es, ahora, FF hacemos:
3003 ? .
? * (imprime el contenido del acumulador
FF
Debemos resaltar que el $60 (RTS=Return To Subroutine) se debe poner siempre al finalizar el programa. Es la manera de decirle al micro la finalización de aquél. De otra forma el control seguirá por las otras direcciones pudiendo ocasionar efectos no deseados.
De igual forma podemos efectuar un programa que cargue el registro X en forma inmediata. Si lo cargamos a partir de la dirección $4000 será:
4000 ? A2 (código de operación de carga inmediata del registro X)
4001 ? 91 (operando)
4002 ? 60
Ejecutándolo de igual manera y comprobando su contenido:
4003 ? .
? R
4003 ? .
? X (imprime el contenido del registro X)
91
Prueben ustedes efectuando el mismo procedimiento para el registro Y. Carguen el programa a partir de la dirección $5FFA.

Modos de direccionamiento del micro 6510

Antes de comenzar a describir los diferentes tipos de direccionamiento, aclaremos algunos criterios de notación. Cuando se representa el contenido de los registros se utilizan los signos (y). Por ejemplo para denotar el contenido del acumulador: (Acc); contenido del registro X: (X); contenido de una dirección determinada de memoria: (M). Cada modo de direccionamiento nos permite acceder, de distintas formas, a la dirección efectiva (D.E.) donde se encuentra el operando buscado. Ellos son:
DIRECCIONAMIENTO INMEDIATO: El operando se encuentra a continuación del código de instrucción. Son necesarios 2 Bytes (uno para el código de instrucción y el otro para el operando). Los casos antes vistos sirven de ejemplos.

Tabla 1

Decimal	Hexadecimal	Binario
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

DIRECCIONAMIENTO ABSOLUTO: La D.E. se encuentra a continuación del código de instrucción almacenada en dos Bytes (primero parte baja y luego parte alta). Por ejemplo si debemos cargar el acumulador con el contenido de la dirección $19A0:
dddd AD (código de instrucción)
dddd+1 A0 (parte baja de la direc.)
dddd+2 19 (parte alta de la direc.)
donde dddd es una dirección inicial. Esta instrucción requiere 3 Bytes.
DIRECCIONAMIENTO PAGINA CERO: Se llama página cero a los primeros 256 Bytes de memoria. Estos están comprendidos entre $0000 y $00FF. Como verán la parte alta de cualquier dirección dentro de este rango es siempre 0. Por ello, en este modo, sólo basta con especificar la parte baja que será la D.E. Es por esta razón que este modo es más rápido (en ejecutarse) que el anterior. Por ejemplo, cargar el acumulador con el contenido de la dirección $00A1:
dddd A5 (código de instrucción)
dddd+1 A1 (sólo la parte baja)
Esta instrucción requiere 2 Bytes.
DIRECCIONAMIENTO INDEXADO: Un valor es almacenado en uno de los registros índices (X o Y).
La d.E. se obtiene sumando este valor a la dirección que se especifica en la instrucción.
Supongamos que (X)=$02, si realizamos:
dddd BD (código de instrucción)
dddd+1 19
dddd+2 30
el acumulador se cargará con el contenido de la dirección dada por $ 3019+(x)=$ 3019+$ 02=$ 301 B (D.E.). Aquí se necesitan 3 Bytes.
DIRECCIONAMIENTO INDEXADO PAGINA CERO: El segundo Byte de la instrucción es sumado al contenido del registro índice especificado (el carry no se propaga al Byte alto de la D.E.), para formar la parte baja de la D.E. La parte alta es forzada a cero. Por ejemplo si (Y)=$0A y:
dddd B6 (código de instrucción)
dddd+1 1F
cargará el registro X con el contenido de la dirección
$1F+(Y)=$1F+$0A=$29. También se requieren de 2 Bytes.
DIRECCIONAMIENTO INDIRECTO INDEXADO: El segundo Byte de la instrucción es sumado al contenido del registro X. El resultado apunta a la dirección en página cero cuyo contenido tiene la parte baja de la D.E. El próximo Byte contiene la parte alta de la D.E. Por ejemplo si (X)=$03 y:
dddd A1 (código de instrucción)
dddd+1 A0
además tenemos que ($00A3)=$10 y ($00A4)=$40, entonces este direccionamiento cargará el acumulador con el contenido de la dirección $4010 (D.E.). Aquí se necesitan 2 Bytes. A este modo también se lo conoce con el nombre de Pre-Indexado.
DIRECCIONAMIENTO INDEXADO INDIRECTO: El segundo Byte de la instrucción apunta a una dirección de página cero. El contenido de ésta es sumado al contenido del registro Y para formar la parte baja de la D.E. Si existe carry, éste se propaga y se suma al contenido de la próxima dirección de página cero. Esto forma la parte alta de la D.E. Por ejemplo si (Y)=$10 y:
dddd B1 (código de instrucción)
dddd+1 E0
y si ($00E0)=$40 y ($00E1)=$BD, el acumulador se cargará con el contenido de la dirección $BD50. Se requieren de 2 Bytes. Este modo también se lo conoce con el nombre de Post-Indexado.

CONOZCAMOS EL MICROPROCESADOR

En esta nota les explicamos cómo es la Unidad Central de Proceso de la Drean Commodore 64 que de alguna manera es similar al cerebro de un ser humano.

El "chip"

La unidad básica en la estructura de todo computador es el chip.
Muchos lectores se preguntarán sobre qué es un chip? pues bien, fisicamente, es una pastilla de silicio con un determinado número de puertas y recibe el nombre genérico de circuito integrado.
En el circuito, las puertas son conformadas por el conjunto de conductores que son recorridos por la corriente eléctrica. (ver figura 1)

C.P.U. de C-64

Como sabemos la CPU es asimilable al "cerebro" de un ser humano capaz de controlar y ejecutar las acciones indicadas por el operador. La unidad central de proceso del equipo C-64 es el microprocesador 6510

Arquitectura de la CPU

Está constituída por los siguientes registros:
Acumulador
Registro Indice X
Registro Indice Y
Registro de estado del microprocesador (P)
Puntero de pila
Unidad de Entrada - Salida
También forman parte de la C.P.U.:
Unidad Aritmética-Lógica (UAL)
Unidad de Control (UC)
Todas estas partes componentes se comunican entre sí, por medio de líneas o buses, que le permiten manejar todo el flujo de datos, señales o "flag" e información.
Asimismo formando parte de esta arquitectura, se encuentra un generador de impulsos de reloj encargado de la sincronización de los ciclos de trabajo del computador.
El conjunto arriba expuesto se denomina Unidad Central de Proceso (CPU). Al estar agrupada en una única pastilla de silicio recibe el nombre de MICROPROCESADOR (Ver fig. 2).

Figura 1

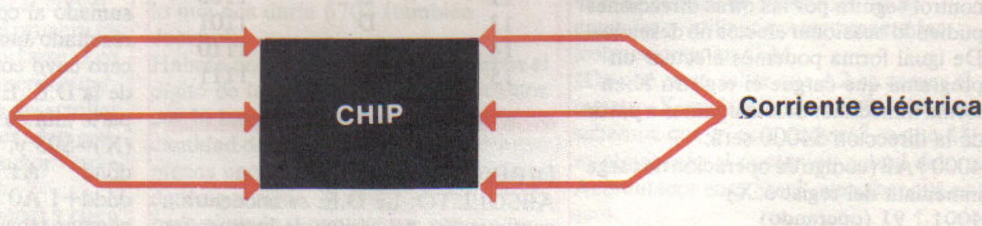
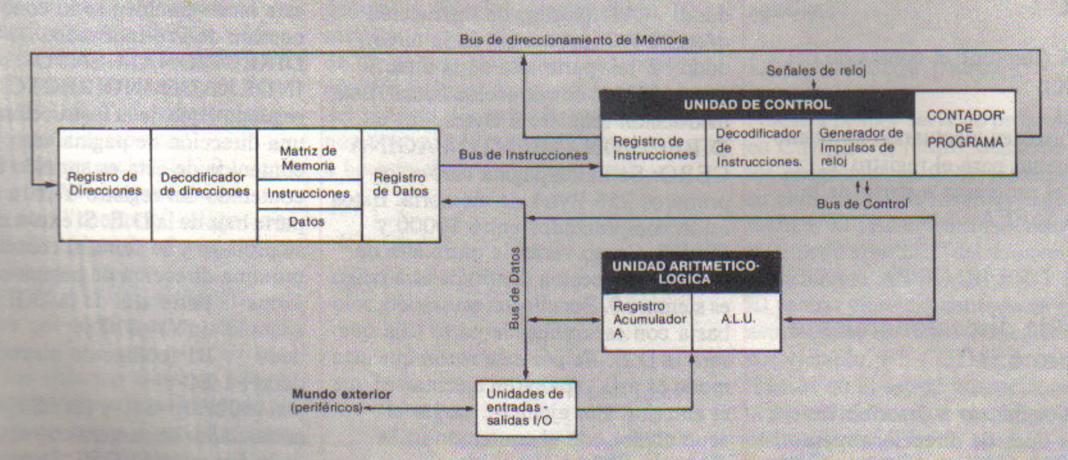


Figura 2



Unidad aritmética-lógica (UAL)

La función de esta unidad es realizar operaciones: aritméticas (sumas, restas) y lógicas (las comparaciones tales como mayor, menor e igual)

Acumulador:

Este es el registro más importante de la C.P.U. Su función es la de almacenamiento temporal de datos. Existen varias instrucciones que nos permite transferir el contenido de direcciones de memoria al acumulador, transferir el contenido del acumulador a direcciones de memoria, modificar el contenido del acumulador o de alguno de los otros registros sin modificar o afectar direcciones de memoria. Es importante resaltar que este registro puede participar directamente en cualquier instrucción lógica o aritmética. Todo lo contrario ocurre con los demás registros (no hay ninguna instrucción que, por ejemplo, nos permita sumar en forma directa los registros X e Y). La figura 3 ejemplifica lo dicho; siempre uno de los operandos proviene del acumulador. El otro puede ser el contenido de algún otro registro o de una posición de memoria determinada.

Registros índice X;Y:

Estos son otros de los registros importantes de la C.P.U. Ellos también almacenan datos (ya sean datos en sí o direcciones de memoria). Permiten, entre otras cosas, el direccionamiento indexado (los datos pueden almacenarse en forma inmediata o en direcciones. En este caso esas direcciones se obtienen sumando a una dirección inicial el contenido del registro X o Y). Para estos registros también existen instrucciones que nos permiten transferir el contenido de una dirección de memoria al X o Y y viceversa.

Puntero de pila

Este registro contiene la primera posición libre en el STACK. El STACK es utilizado para almacenamientos temporales por programas en absoluto y por la C-64.

Unidad de Entrada-Salida

Este registro es utilizado para la administración de memoria del C-64.

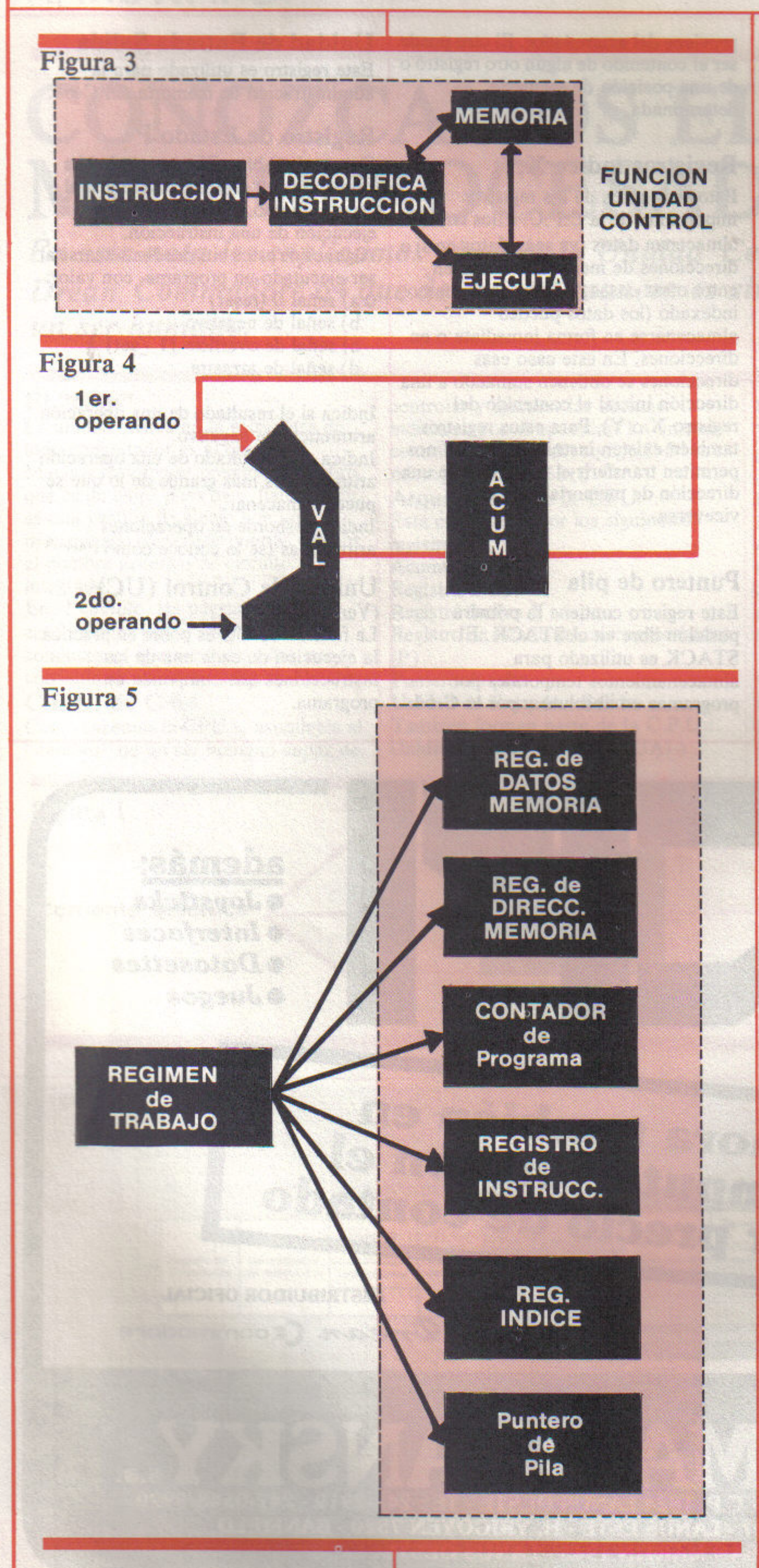
Registro de Estado P

Consta de 8 bits. El número de bits contenidos en este registro indica el estado interno del CPU, luego de la ejecución de una instrucción.
Algunos de estos bits pueden setearse al ser ejecutado un programa, con valor:
0 a) señal 0 (reset)
b) señal de negativo
c) señal de overflow (1 - set)
d) señal de arrastre

Indica si el resultado de una operación aritmética fue negativo.
Indica si el resultado de una operación aritmética es más grande de lo que se puede almacenar.
Indica desborde en operaciones aritméticas (se lo conoce como carry)

Unidad de Control (UC)

(Ver figura 4)
La función de UC es poner en práctica la ejecución de cada una de las instrucciones que componen un programa.



El mecanismo que cumple la UC es:
1º realiza la captura de la instrucción, almacenando en un registro de trabajo la dirección de memoria.
2º en el caso de Basic, decodifica la sentencia y determina el código de operación. Si la orden está relacionada con la Memoria Central, efectúa la transmisión a través de los buses internos, así como también en el caso de efectuar operaciones aritméticas-lógicas.
3º Ejecución propiamente dicha. Para lograr esto, la UC hace uso de la memoria ROM siendo leídas y ejecutadas las instrucciones grabadas en esta última. Tengamos presente que la memoria ROM es escrita de fábrica con un conjunto de microinstrucciones.

Registro de trabajo

(Ver figura 5)
La función de los registros de trabajo es servir de colaboradores, aportando el almacenamiento temporario de datos, operando o direcciones utilizados por la UC o la UAL.
Los registros de trabajo del 6510 son:
- Registro de datos de Memoria (MDR)
- Registro de Direcciones de Memoria (MAR)
- Contador de Programa (PC)
- Registro de Instrucciones (IR)

Función de los registros especiales de la CPU

Registro de datos de Memoria (MDR)
Este registro la función que cumple es la de almacenar los datos, operandos, a fin de ser luego decodificado y ejecutado

Registro de Direcciones de Memoria (MAR)

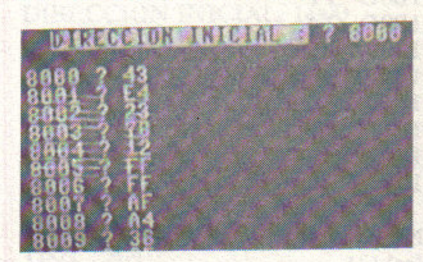
Este registro es el encargado de almacenar las direcciones correspondientes a una instrucción o dato, reservado en la memoria.

Contador de Programa: (PC)

La función que cumple es almacenar la dirección de memoria de la próxima instrucción a ejecutar
La diferencia entre el MAR y el PC radica fundamentalmente en que el primero almacena direcciones de datos, en cambio el PC solamente de instrucciones.
Además el contenido de MAR será siempre el resultado de la transferencia de la dirección almacenada en PC.
Esta comunicación entre el PC y el MAR se logra mediante un bus de direcciones.
La descripción de los restantes registros de trabajo y demás partes del computador serán desarrolladas en próximas notas.

ASSEMBLER 1.1

Continuamos describiendo los comandos del programa de Cristian Parodi para la Drean Commodore 64



Comando Modifica:
Función: Modifica el contenido de la dirección hexadecimal deseada.
Formato: Mdddd/hh
donde: dddd: dirección a modificar
hh : nuevo valor
En nuestro ejemplo, supongamos que debemos modificar el contenido de la dirección 1B04 por 01.
(el PC estaba en 1B07)
1B07 ? . (cr)
? M1B04/01 (cr)
DATO ANTERIOR: A9 si el formato anterior es correcto, el editor imprimirá el dato anterior y el actual de la dirección modificada. Luego volverá a modo carga.
DATO ACTUAL: 01
1B07 ? .
? M2/A (cr) no se respetó el formato del comando
ERROR DE COMANDO
1B07 ? . (cr)
? M0002/0A (cr)
DATO ANTERIOR: C2
DATO ACTUAL: 0A
1B07 ?

Comando Delete:
Función: Este comando borra el contenido de la dirección deseada poniendo el código de operación de la instrucción NOP en esa dirección (NOP= no operation)
Formato: Ddddd
donde: dddd es la dirección a borrar (su contenido)
En nuestro ejemplo, borraremos la dirección 1B05
1B07 ? . (cr)
? D1B05 (cr)
DIRECCION 1B05 BORRADA
1B07 ? .
? L1B05 (cr)
1B05 EA (código de operación de NOP)
1B07 ?

Comando Cambia:
Función: Cambia el valor del PC y/o de la dirección inicial ingresada al comienzo del programa.
Formato: Cdddd
donde dddd: nuevo valor del PC
Este comando nos permite movernos libremente a través de toda la memoria del C-64. Suele suceder, a veces, que el programa y sus datos se almacenan en áreas de memoria distintas y no necesariamente seguidas. Con este





comando podemos cargar el programa principal y luego cambiar el contenido del PC para comenzar a ingresar los datos antes mencionados. Para ejemplificar lo dicho, carguemos un programa cualquiera a partir de la dirección hexadecimal 1B00 y los datos de él a partir de la dirección hexadecimal 3000:

```
DIRECCION INICIAL : ? 1B00
1B00 ? A9 (cr)
1B01 ? 0D (cr)
1BO2 ? 20 (cr)
1B03 ? D2 (cr)
1B04 ? FF (cr)
1B05 ? A2 (cr)
1B06 ? 0   (cr)
1B07 ? BD (cr)
1B08 ? 0   (cr)
1B09 ? 30 (cr)
1B0A ? 20 (cr)
1B0B ? D2 (cr)
1B0C ? FF (cr)
1B0D ? E8 (cr)
1B0E ? E0 (cr)
1B0F ? 9   (cr)
1B10 ? D0 (cr)
1B11 ? F5 (cr)
1B12 ? 60 (cr)
1B13 ? .   (cr)
     ? C3000 (cr)
NUEVA DIRECCION INICIAL :
3000 (1)
3000 ? 20 (cr)
3001 ? 20 (cr)
3002 ? 20 (cr)
3003 ? 20 (cr)
3004 ? 48 (cr)
3005 ? 4F (cr)
3006 ? 4C (cr)
3007 ? 41 (cr)
3008 ? OD (cr)
3009 ? .   (cr)
     ? L   (cr)
3000   20
3001   20 como se cambió la direc.
3002   20 inicial, el comando L lista
3003   20 desde la 3000 hasta (PC)-1
3004   48 (como el PC quedó en 3009,
3005   4F se lista hasta la 3008)
3006   4C
3007   41
3008   OD
3009 ? . (cr)
     ? C1B13 (cr)
NUEVA DIRECCION INICIAL : ?
1B00 (cr) (2)
1B13 ? . (cr)
     ? L
1B00   A9
1B01   OD
1B02   20
1B03   D2 ahora el L lista desde la
1B04   FF direc. inicial actual (1B00)
1B05   A2 hasta (PC)-1 (acabamos de
1B06   00 cambiar el PC a 1B13, es
1B07   BD decir que se imprime hasta la
1B08   00 1B12)
1B09   30
1B0A   20
1B0B   D2
1B0C   FF
1B0D   E8
1B0E   EO
1B0F   09
1B10   DO
1B11   F5
1B12   60
1B13 ? . (cr)
     ? C3000 (cr)
NUEVA DIRECCION INICIAL : ?
(cr) (3)
3000 ? . (cr)
     ? L (cr)
(lista los contenidos de las direcciones
1B00 hasta la 3000)
 '   el editor interpreta que la direc.
 '   inicial no ha sido cambiada;
 '   sigue en 1B00
3000 ?
```

Lo que hace este programa es escribir la palabra "HOLA" en la pantalla. Primero se cargó el programa en si (direcciones hexadecimales 1B00-1B12) y luego se ingresaron los códigos ASCII de los caracteres a imprimir (direcciones 3000-3008). Al comienzo se cargó el programa en la forma ya conocida, es decir instrucción a instrucción. Al finalizar esta tarea, cambiamos el contenido del PC al valor 3000 y empezamos a cargar los datos (ver (1)). Siempre que se cambie el PC, el editor preguntará si deseamos cambiar también la dirección inicial actual. Podemos optar por cambiarla o por dejarla con el valor actual. Lo primero se efectúa introduciendo el nuevo valor (ver (1), (2)). Lo segundo se realiza oprimiendo (cr) simplemente. De esta forma el editor no alterará la dirección inicial actual.

Se debe recordar que siempre que cambiemos el PC y/o la dirección inicial, los comandos van a operar en forma relativa a ellas. Es decir que el comando L imprimirá otros datos si la dirección inicial fue modificada al igual que el PC (en nuestro ejemplo esto sucede). Algo similar ocurre con el comando R, ya que éste ejecuta el programa almacenado desde la dirección inicial actual. Esto debe tenerse muy en cuenta para evitar sorpresas desagradables.

Si el formato del comando C no es la correcta, el editor imprimirá el mensaje correspondiente. En caso concreto, el editor volverá a modo carga.

Comando Stop:
Función: Para la ejecución del editor. Este puede reinicializarse a través del comando BASIC CONT. En este caso el editor volverá a modo carga imprimiendo el valor actual del PC (que es igual al que tenía antes de efectuarse en comando Stop)
Formato: S
Para el ejemplo antèrior sería (recordar que el PC estaba con 3000)

```
3000 ? . (cr)
     ? S (cr)
BREAK IN 48
READY
CONT (cr)
3000 ? (el editor queda nuevamente a la
        espera de una instrucción o de un
        comando).
```

---()---

Comando Bye:
Función: Reinicializa la C-64 (se imprime el mensaje que aparece cuando prendemos la máquina por primera vez). Cabe acotar que todos los datos y programas (incluyendo el editor) se borran cuando este comando es solicitado.
Formato: B
Volviendo a nuestro ejemplo:

```
3000 ? . (cr)
     ? B (cr)
```

(aparece el mensaje antes citado)

Comando X:
Función: Imprime en pantalla el contenido del registro X.
Formato: X

---()---

Comando Y:
Función: Imprime en pantalla el contenido del registro Y.
Formato: Y

---()---

Comando P:
Función: Imprime en pantalla el contenido del registro de estado del microprocesador (Processor status register)
Formato: P

---()---

Comando *:
Función: Imprime en pantalla el contenido del acumulador.
Formato: * (asterisco)
Ejemplo: El siguiente programa carga el acumulador con AA, el registro X con BB, el registro Y con CC e imprime el contenido actual del registro SP (para aquellos que no conocen el lenguaje máquina de la C-64, este registro contiene información del esto del procesador en un momento dado; indica cuándo hubo carry u overflow o cuándo una operación aritmética tuvo respaldo negativo. Con esto quiero decir que el contenido de este registro varía constantemente).

```
DIRECCION INICIAL : ? 1B00
1B00 ? A9
1B01 ? AA (cr)
1B02 ? A2 (cr)
1B03 ? BB (cr)
```

1B04 ? AO (cr)
1B05 ? CC (cr)
1B06 ? 60 (cr)
1B07 ? . (cr)
? R (cr) (ejecutamos el programa)
1B07 ? . (cr)
? *(cr)
AA (contenido del acumulador)
1B07 ? . (cr)
? X (cr) (contenido del registro X)
BB
1B07 ? . (cr)
? Y (cr) (contenido del registro Y)
CC
1B07 ? . (cr)
? P (cr)
B1 (contenido del registro SP)
1B07 ? (el editor se queda a la espera de un dato o de un comando)

---()---

Comando Save:
Función: Almacena en el disket el programa que se encuentra en memoria desde la dirección inicial actual hasta la dirección representada por (PC)-1.
Formato:↑Para mostrar mejor este comando, vamos a almacenar el programa anterior:
(el editor estaba a la espera en la direc. 1B07)
1B07 ? . (cr)
? ↑ (cr)
NOMBRE DEL FILE:? PRUEBA (cr)
SAVING: PRUEBA
DIRECCION INICIAL: 1B00
DIRECCION FINAL : 1B06
1B07 ? (editor a la espera)
Luego de introducir el comando, el editor nos interrogará sobre el nombre de nuestro programa. Este debe ser, en longitud, menor o igual a los ocho (8) caracteres. Caso contrario se volverá a preguntar por el nombre del programa. Si el nombre del programa cumple con las condiciones dadas, el editor imprimirá tres mensajes: el primero nos indica que el programa se está grabando, los otros dos corresponden a la dirección inicial y final de nuestro programa. Luego de haber efectuado este comando el editor regresa al modo carga.

---()---

Comando Load:
Función: Almacena en memoria el programa guardado en el diskett.
Formato: ←
Para mostrar su funcionamiento, carguemos el programa PRUEBA
DIRECCION INICIAL: ? 4ADD (cr)
4ADD ? . (cr)
? ← (cr)
NOMBRE DEL FILE:? PRUBA (cr)
FILE NOT FOUND
4ADD ? . (cr)
? (cr)
NOMBRE DEL FILE: ? PRUEBA (cr)
LOADING: PRUEBA
DIRECCION INICIAL: 1B00
DIRECCION FINAL: 1B06
4ADD ? . (cr)
? C1B07 (cr)
NUEVA DIRECCION INICIAL: ? 1B00 (cr)
1B07 ? . (cr)
? L (cr)
1B00 A9
1B01 AA
1B02 A2
1B03 BB
1B04 AO
1B05 CC
1B06 60
1B07 ? (editor a la espera)
Como se verá, al principio ingresamos un nombre erróneo. Luego como el editor no encontró en el directorio del disket el programa PRUBA imprimió el mensaje correspondiente y regresó a modo carga. En el segundo intento ingresamos el nombre correcto (aquí también el nombre debe ser menor o igual a los ocho caracteres) y se nos confirmó la carga del programa a través del mensaje "LOADING...". También se nos indicó la dirección inicial y final del mismo. Luego procedimos a listarlo con lo que tuvimos que cambiar el valor del PC y la dirección inicial para que ello ocurra. El objetivo de informar sobre la dirección inicial y final es para que no tengamos que llevar aparte un registro de las direcciones que ocupa un programa determinado. Por ello se ingresó 4ADD en la dirección inicial (valor arbitrario). Luego cambiamos el PC a la dirección siguiente a la final (si hubiese sido la dirección final, el dato que allí teníamos se habría perdido).
Con este comando se termina la descripción de los de CLASE I.
A continuación los comandos CLASE II.

---()---

Comando: @
Función: Imprime en pantalla el directorio del diskett,
Formato:

---()---

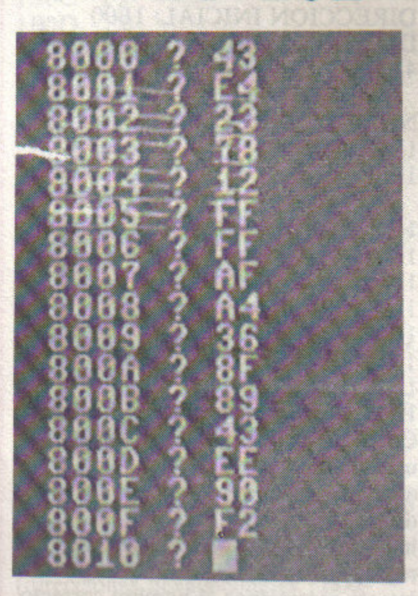
Comando @ V:
Función: Envía al Disk Drive el comando VALIDATE. El editor imprime el mensaje "REORGANIZANDO EL DIRECTORIO" cuando este comando es solicitado.
Formato: @ V
Comando @ I:
Función: Envía al Disk Drive el comando INITIALIZE.
Formato: @ I
Comando @ S:
Función: Borra del disket el programa seleccionado.
Formato: @ S
Para mostrar este comando, borremos el programa PRUEBA.
DIRECCION INICIAL: ? 2000
2000 ? . (cr)
? @ S (cr)
NOMBRE DEL FILE:? PRUEBA (cr)
FILE PRUEBA BORRADO
2000 ? (editor a la espera)
Si por error introducimos este comando, podemos irnos de él ingresando simplemente un ".". En caso que efectivamente querramos borrar un programa, el editor nos informará si ha ejecutado esa orden a través del mensaje "FILE... BORRADO"
Luego que el editor halla culminado con alguno de los comandos pertenecientes a la CLASE II, éste volverá al modo carga imprimiendo el contenido actual del PC.

Comentarios

i) En caso de ingresar un comando no descripto en este apunte, el editor responderá con el mensaje: "COMANDO DESCONOCIDO"
Luego volverá a modo carga.
ii) Una vez cargado el editor en memoria, ésta queda distribuída de la siguiente forma:
MEMORIA LIBRE: $ 1B00 $ 9FFF
TEXTO BASIC (EDITOR): $ 0801 $ 1A78
VARIABLES DEL EDITOR:
$ 1A79 $ 1AF0
$ 0000 $ CFFF
$ 0002
Las posiciones de memoria $ C2FF-$ CAFF se utilizan para almacenar los caracteres de las variables string que usa el editor. Esto se hizo para que se disponga de esa área de memoria libre. De otra manera, el intérprete BASIC hubiese utilizado las últimas posiciones de la RAM BASIC (es lo que hace normalmente) con lo que se destruiría todo programa (también las mismas variables) allí almacenado.
A partir de la dirección $ C000 se encuentra el programa que realiza lo descripto anteriormente (actualiza el puntero de comienzo de variables string). Este programa se ejecuta continuamente y no modifica para nada el normal funcionamiento del editor o de los programas cargados. Además, su tiempo de ejecución puede considerarse despreciable.
La posición de memoria $ 0002 contiene un dato que es utilizado por el programa anterior.

ASSEMBLER 1.1

Debido a la modularidad de este software, se pueden quitar o agregar comandos. Si no se dispone del drive, se podrán cambiar esos comandos permitiendo el trabajo con cassettes. En esta última parte explicamos cómo se puede modificar.



Con este número culminamos con la presentación del programa ASEMBLER 1.1. Aquellos que lo hayan analizado habrán notado que, debido a la modularidad de éste, es posible modificarlo. Es decir que se pueden quitar o agregar comandos. Por ejemplo, si no se dispone del drive, los comandos referidos al uso de éste no tienen sentido. Es por ello que podemos cambiarlos por comandos que nos permitan trabajar con cassette o, simplemente, borrar las subrutinas correspondientes. Si efectuamos esto último la memoria libre del editor ($1B00-$9FFF) aumentará ya que el texto Basic se achica. Para determinar la nueva área, efectúen el siguiente procedimiento:

1) Listen las direcciones $002D-$002E. Los contenidos de éstas representan la dirección de comienzo del área de almacenamiento de las variables que utiliza el editor (Byte alto y bajo respectivamente).

2) Sumen el valor de $78 a esta dirección. El resultado representa el inicio del área libre. Como verán es menor que $1B00. El tope sigue siendo $9FFF.

En caso de agregar un nuevo comando efectúen el mismo procedimiento. La única diferencia es que la dirección de inicio será mayor que $1B00.

Aquí les ofrecemos un nuevo comando que consideramos útil para trabajar con el KERNAL de la C-64. Se deberá insertar al programa las líneas que se detallan en el recuadro.

Comando Transfiere:

Función: Transfiere el bloque de memoria deseado a partir del contenido actual del PC.
Formato: Tcccc-ffff
donde
cccc,ffff: Dirección de comienzo y fin del bloque de memoria a transferir.
Nota: En caso en que la dirección representada por cccc sea mayor que la dirección ffff, el editor imprimirá:

ERROR DE COMANDO

Luego volverá a modo carga.

Para ejemplificar el uso de este nuevo comando, tratemos de modificar la rutina que limpia la pantalla y pone el cursor en el borde izquierdo superior. Esta rutina se encuentra almacenada en ROM (Read Only Memory) por lo que nos es posible alterarla. Es por ello que debemos transferirla a la memoria RAM (Random Access Memory):

DIRECCION INICIAL: ? 3000
3000 ? . (cr)
? TE544-E598 (cr) Direcciones
? de comienzo y fin de la rutina.
3055 ? luego de la transferencia (en realidad es una copia) el editor queda a la espera. Si listamos esta área nos encontramos con dicha rutina.

ADVERTENCIA: 1) La transferencia del bloque deseado siempre se realizará a partir del contenido actual del contador de programas (PC).
2) No se deben transferir bloques cuyos tamaños superen al área de memoria libre. De otra forma puede suceder que la C-64 se "cuelgue".

Listado Comando T

```
 45 IFMID$(A$,1,1)="T"THENGOSUB342:GOTO30

341 REM *** COMANDO TRANSFIERE ***
342 IFLEN(A$)<>10THENGOSUB1010:RETURN
346 M$=MID$(A$,2,4):CO$=MID$(A$,7,4)
347 IFCO$<M$THENGOSUB1010:RETURN
348 A$=M$:GOSUB500:IFF=1THENGOSUB1010:RETURN
349 M=P:A$=CO$:GOSUB500:IFF=1THENGOSUB1010:RETURN
358 N=P:R=0:FORI=MTON:POKEAD+R,PEEK(I):R=R+1:NEXTI:PRINT:AD=AD+R:
    RETURN
```

EL ARTE DE EDITAR

Un tema muy importante, que desarrollan J.D. Willis y M. Miller en su "Guía práctica para el conocimiento del Commodore 64" (ediciones Aura), que anticipamos en estas páginas.



Si usted es un mecanógrafo experimentado y posee su propia máquina de escribir, ¿por qué tendría que aprender a usar el procesador de textos? Si usted mecanografía documentos que se usan solamente una vez y usted siempre los escribe perfectamente, tiene muy poca necesidad de aprender el tratamiento de textos. Si usted comete errores o tiene que mecanografiar la misma cosa varias veces (por ejemplo, una carta que hay que mandarla a varias personas), o si tiene que revisar y volver a escribir documentos, el procesador de textos le puede ayudar a hacer más trabajo en menos tiempo.

Uno de nuestros amigos es presidente de un comité que cada año adjudica ocho becas para estudiantes y recibe entre 50 y 150 solicitudes. Cada año tiene que escribir ocho cartas felicitando a los que han sido elegidos para la beca, otras ocho cartas a los candidatos que aún pueden recibir la beca, si alguno de los ocho primeros decide no utilizarla y una cantidad de cartas a los solicitantes que no han sido elegidos. Tiene tres tipos de cartas estándar, que modifica ligeramente cada año. Antes de haber empezado a utilizar el procesador de textos, él o su secretaria tenían que mecanografiar cada carta, comprobar si no contenía errores, volverla a escribir si los contenía, y finalmente firmar y mandar las cartas por correo. Con el procesador de textos, le manda al ordenador cargar el fichero que contiene la carta-modelo en la memoria. El ordenador exhibe el texto de la carta en la pantalla, él lo edita para que refleje las circunstancias particulares del concurso del año actual y añade cualquier nota personal que quiera. Luego teclea la dirección correcta y apreta la tecla que indica al ordenador que imprima la carta en una impresora de calidad, que tiene los tipos de letra iguales que una máquina de escribir. Prácticamente, con todos los programas de Tratamiento de Texto se puede electrónicamente insertar o eliminar material. Si usted teclea «Tkhis», cuando quería «This», fácilmente puede desplazar el cursor encima de la letra «k» (con las teclas de control de movimiento del cursor, las que tienen flechas encima). Aprieta la tecla DEL y la letra "k" desaparece. El procesador también puede eliminar un espacio en blanco; en lugar de «T his» aparece «This». Muchos procesadores de textos le permiten eliminar el material letra por letra. Algunos permiten eliminar palabras o frases enteras o incluso todo un párrafo, apretando una o dos teclas. La función de insertar funciona igual que la de eliminar. Si usted teclea «Su solicitud es una de mejores» y luego quiere añadir «las», es una cosa simple. Usted desplaza el cursor encima de la letra «m» en la palabra «mejores», aprieta la tecla INSERT. La mayoría de los procesadores entonces crea una «ventanilla» en ese sitio en la pantalla y usted puede teclear lo que quiera. Cuando haya terminado, aprieta la tecla INSERT otra vez y el procesador cierra la «ventanilla» en la que se encuentra el nuevo material. La misma operación en la máquina de escribir exigiría volver a mecanografiar toda la carta.

Nosotros opinamos que la habilidad de insertar o eliminar material electrónicamente hace más que sólo aumentar la velocidad de escribir. Ya que el proceso de editar y cambiar el material es difícil en la máquina de escribir, nuestros manuscritos se convierten en un mar de marcas de lápiz rojo. Flechas rodeando todo un párrafo que apuntan hacia el sitio donde este material debe trasladarse en la próxima versión. El esbozo empieza a ser inutilizable, ya que es difícil de seguir todos los cambios y correcciones marcadas en él. Y la idea de volver a mecanografiarlo es muchas veces rechazada por falta de tiempo. Con el procesador de textos las correcciones se efectúan al momento.

Si usted quiere trasladar un párrafo de la página 2 en medio de la página 8, la mayoría de los programas le permite marcar el párrafo e indicar al ordenador que lo traslade al nuevo sitio. Inserciones, eliminaciones y traslados de bloques enteros se solucionan usando una o dos teclas y se puede ver el resultado inmediatamente. Algunos programas incluso tienen una instrucción «undo» que le permite volver atrás a la versión como estaba antes de empezar a hacer los cambios; en caso de que usted la encuentre mejor que la versión nueva. Somos capaces de hacer los repasos más rápidamente y podemos perfeccionarlos más, ya que el trabajo manual aburrido ha sido

reducido. Ahora uno se puede concentrar en el proceso de edición más que volver a teclear los 90% sólo porque se han hecho cambios en los 10% restantes. La frase que ha sido escrita correctamente la primera vez no necesita ser rehecha aunque el documento pase por muchas revisiones.

Requerimientos básicos para fácil Tratamiento de Textos

Al 64, aunque es un ordenador excelente, le faltan algunas características que serían necesarias para un tratamiento de textos profesional. Algunas características deseables para un buen tratamiento de textos están descritas aquí.

Pantalla de gran capacidad. Cuanto más se puede ver en la pantalla con una mirada, más fácil es componer y editar un documento. Un buen procesador de textos tiene por lo menos un «display» de 24 líneas por 80 caracteres. Además tiene que tener códigos especiales de impresión que subrayan, resaltan palabras, etc., y lo más importante, tiene que ser capaz de editar sofisticadamente.

Las pantallas de color verde o ámbar cansan menos los ojos que las estándar blanco-negras o las pantallas de televisor de color. El ámbar es especialmente buen color y en muchos países europeos es exigido en los ordenadores de oficina.

La pantalla del Commodore 64 con sus 25 líneas de 40 caracteres se puede utilizar en tratamiento de textos, pero uno se cansa rápido de ella. Accesorios que le permiten una capacidad más grande (ver capítulo 10) cuestan varios cientos de dólares. Eso no significa que la pantalla no se pueda usar. Cientos de miles de usuarios de Apple II, por ejemplo, escriben alegremente en sus máquinas con la pantalla de sólo 24 líneas por 40 caracteres. El formato 25 x 80 requiere ya un video-monitor más costoso. Si usted usa el televisor con formato 25 x 80, las letras son demasiado pequeñas y borrosas para una lectura confortable.

Un teclado con muchas teclas funcionales. Cuantas más teclas funcionales, más fácil es aprender y usar un programa de tratamiento de textos. Se pueden asignar las funciones especiales (como eliminar o trasladar material) a una tecla funcional. Sólo con apretar la tecla para una operación determinada ésta se ejecuta. Sin las teclas funcionales habría que recordar una rutina complicada. Por ejemplo, para archivar algo que uno ha escrito en el TRS-80 Modelo III que utiliza el procesador de textos «Scripsit», hay que apretar primero la tecla BREAK. Luego hay que teclear una «S», un espacio y el nombre que se quiere asignar al documento. Luego apretar la tecla RETURN. Cada una de las operaciones requiere varios pasos, ya que el ordenador no tiene teclas funcionales programables. En el Eagle uno aprieta la tecla SAVE y el ordenador pregunta qué nombre hay que asignar al documento. Se teclea el nombre y aprieta RETURN. Hay que recordar menos y hay menos teclas que apretar.

El teclado del Commodore 64 no está mal. Tiene un buen toque y le permite teclear mayúsculas y minúsculas (algunos ordenadores sólo admiten mayúsculas). Con sus cuatro teclas funcionales programables es mejor para el tratamiento de textos que teclados como el de Apple II. Cada una de estas teclas puede tener dos funciones asignadas (una para el modo normal, otra para el modo con tecla de cambio), junto con varias teclas de función preprogramadas para el tratamiento de textos. Con más teclas funcionales y con cuatro teclas de control de movimiento de cursor en lugar de dos, el teclado del Commodore 64 sería un teclado mejor para tratamiento de textos.

Memoria secundaria rápida, de alta capacidad y fiabilidad. Los programas de tratamiento de textos para el Commodore 64 utilizan el cassette o la unidad 1541 de diskette para archivar datos y ficheros de tratamiento de textos. El sistema de casette es muy lento y tiene una capacidad limitada. El 1541 diskette es una de las unidades de diskettes más lentas que hemos usado, aunque se rumorea que una versión mucho más rápida está a punto de salir (no estaba disponible cuando se escribió este libro). La unidad 1541 puede archivar como máximo algo más de 170.000 caracteres en cada diskette. Los anuncios de Commodore rezan que es una memoria «rápida y de alta capacidad». Es rápida y de alta capacidad sólo en comparación con el sistema de casette. Otros sistemas de diskettes le permiten grabar hasta 700.000 caracteres y trabajan mucho más rápidos que la 1541.

La unidad 1541 no es rápida ni de alta capacidad y le da muy mal servicio a quienes dicen que lo es. No obstante, la 1541 es una unidad de alta fiabilidad, una de las más fiables que nunca hemos usado. Y es una de las menos costosas unidades de diskette para ordenadores personales. Se puede aguantar más una unidad de diskette que es lenta pero fiable que una que es rápida, de alta capacidad pero poco fiable. A nosotros nos gusta la 1541, pero no por razones por las que lo anuncia el Commodore.

Mucha memoria RAM. Eso puede o no ser problema para usted si usted escribe material que no es más largo que una carta comercial, no necesitará mucha memoria para registrar el documento a medida que lo va tecleando. No obstante, si escribe informes, artículos para revistas o libros, una memoria RAM limitada es muy inconveniente. En este aspecto el 64 obtiene puntos positivos. El 64-estándar tiene más memoria RAM accesible para el usuario que cualquier otro ordenador de su nivel de precio. Si el programa de tratamiento de texto está situado en el cartucho ROM, toda la RAM disponible se puede utilizar para registrar el documento que uno está escribiendo. Aunque su programa de Tratamiento de Texto estaría en el casette o diskette y así tendría que cargarse en la RAM, quedaría bastante memoria disponible.

El 64, igual que otros ordenadores Commodore, usa unidades de diskette «inteligentes». Eso significa, que no hace falta cargar un programa llamado DOS (sistema operativo de diskette) en la memoria RAM, si uno tiene intención de usar el 64 con la unidad de diskette 1541. Algunos ordenadores, como el Apple II, utilizan las unidades de diskettes estándar, que requieren cargar el DOS en la RAM, antes de usar el diskette. El DOS ocupa miles de caracteres de la memoria. El 64 puede tener aproximadamente 64.000 caracteres de ROM y RAM. Los ordenadores más costosos como el IBM-PC aceptan más de 500.000 caracteres de memoria, pero no se puede comprar un IBM-PC por menos de 400 dólares.

Ausencia de Glitchitis. Glitchitis es una enfermedad normalmente encontrada en todos los tipos de equipamiento electrónico. Generalmente ataca, cuando un mal funcionamiento puede ocasionar más daños, frecuentemente ocurre por causas desconocidas, y no se puede repetir, cuando uno quiere demostrar qué es lo que ha pasado. Aún peor, casi nunca ocurre cuando uno está intentando explicar el problema a alguien que podría arreglarlo. Generalmente se repite, sin embargo, cuando uno está usando el ordenador para una cosa muy importante (por ejemplo, tecleando un informe a la una de la madrugada que tiene que ser entregado a las nueve de la mañana siguiente).

El 64 parece notablemente libre de esta enfermedad, aunque algunos de sus

competidores están especialmente propensos a ella. Con excepción de algunas dificultades con interferencias en modelos más antiguos, el 64 disfruta de una reputación de una operación fiable. Hay otras características que uno puede buscar a la hora de comprar un procesador de textos profesional, pero los arriba mencionados son algunos de los más importantes.

Características concretas deseables en el tratamiento de textos

Cientos de programas de tratamiento de textos para pequeños ordenadores están disponibles por el precio entre 10 y 1.000 dólares, programas para el Commodore 64 están más bien en la parte más baja de la escala. La mayoría van por menos de 100 dólares y dado que muchos programas nuevos constantemente aparecen en el mercado, quizás algunos de los buenos programas no serán mencionados aquí. Por esta razón le intentamos dar una corta lista de características que hay que buscar en un buen procesador de textos.

Modalidad DELETE

¿Cuán fácil es eliminar un texto de la pantalla? Algunos programas simples sólo permiten eliminar material caracter por caracter, otros muchas opciones, incluyendo:

- Eliminar un caracter.
- Eliminar una palabra. Eliminar material separado por caracteres en blanco en ambos extremos.
- Eliminar una frase. Elimina desde la mayúscula precedida por un espacio en blanco hasta el punto final u otro caracter de puntuación.
- Eliminar una línea. Elimina una línea de texto en la pantalla. Un sustituto pobre de eliminar la frase, ya que las líneas en la pantalla pocas veces coinciden con la longitud de una frase.
- Eliminar una parte de una línea. Por ejemplo, elimina el texto desde el cursor hasta el final de la línea.
- Eliminar un párrafo. Elimina desde la primera frase marcada hasta la última frase marcada.
- Eliminar un bloque. Ver la sección sobre bloques para una explicación
- Eliminar todo. Elimina todo que lo haya en la memoria del ordenador.
- Eliminación - volver atrás (undo).
- Verificación de una eliminación. Cuando es instruído para eliminar mucho material, el programa controla si es realmente cierto, preguntando: ¿De verdad? S/N o «Apriete el RETURN para eliminar el documento , antes de proceder a la ejecución.

Modalidad INSERT

Métodos para añadir o insertar material.

- Un insert destructivo. Reemplaza el material tecleando el nuevo sobre el anterior.
- Insert no destructivo. Corre el material hacia atrás al introducir el material nuevo o crea una «ventanilla— de espacios en blanco en la cual se puede introducir nuevo material.

Modalidad de BLOQUES

Un bloque es una sección del texto que ha sido marcada, por ejemplo apretando la tecla con flecha abajo al principio y al final de la misma. Luego se efectúan varias operaciones en este bloque de texto. Se pueden trasladar, eliminar o duplicar bloques de texto. Duplicándolo se introduce el mismo material en varios sitios del texto. Informes, por ejemplo, a menudo contienen muchas tablas complicadas cuyo formato es el mismo pero los números son los mismos. La tabla se introduce una sola vez, se marca como bloque de texto y se inserta en cualquier sitio en el texto. Una vez que la tabla esté en su lugar, sólo se rellena con números.

- Eliminar un bloque. ¿Se puede eliminar un bloque?
- Desplazar un bloque. ¿Se puede mover un bloque de un sitio a otro?
- Duplicar un bloque. ¿Se puede duplicar un bloque en otro sitio del texto, sin que se elimine de su lugar original?
- Bloques múltiples. ¿Se puede marcar más que un bloque o existe la limitación de marcar solamente un bloque al mismo tiempo?

Control de Cursor

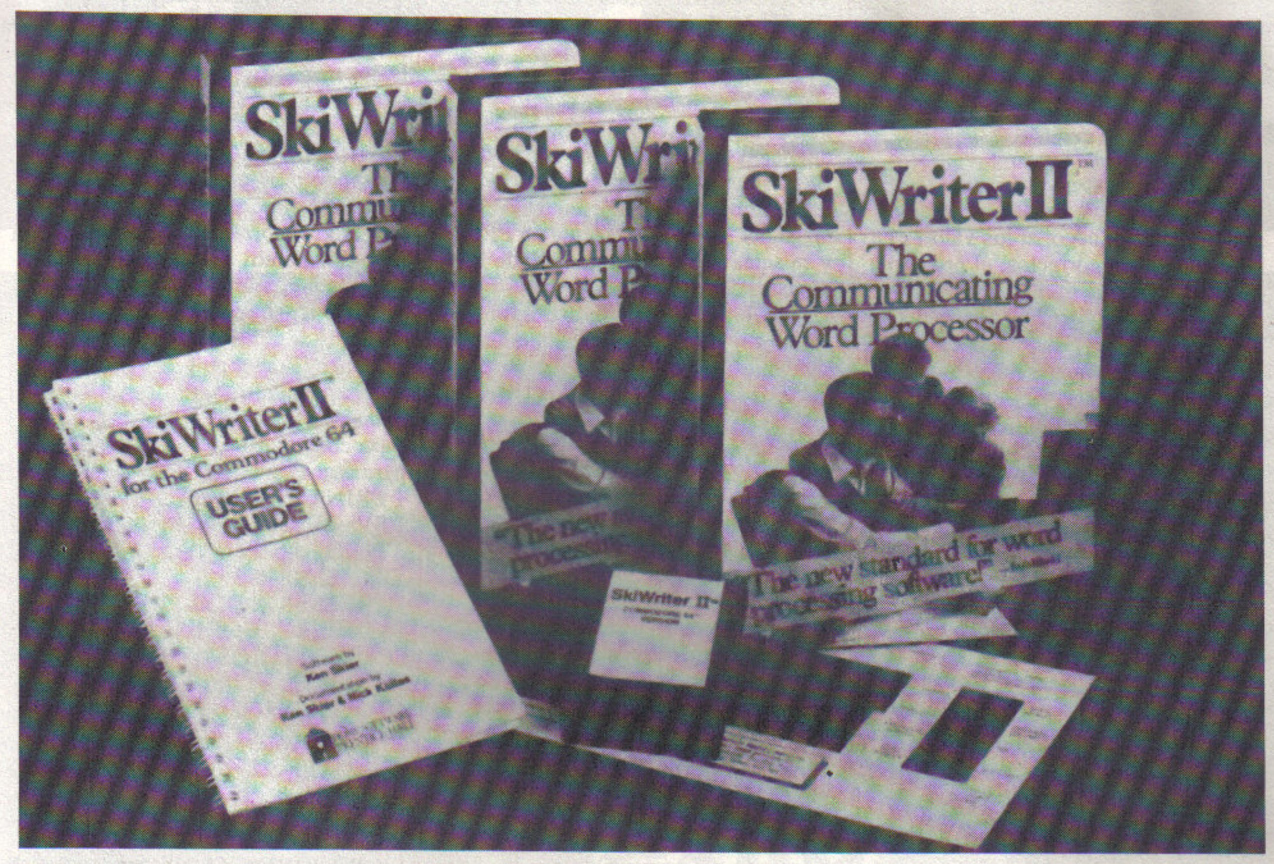
El cursor es un rectángulo o bien sólido o bien parpadeando que le indica donde el material que usted teclea aparece en la pantalla. También le indica qué es lo que será afectado por la instrucción de eliminar o insertar material, por ejemplo, en la mayoría de los programas, hay que desplazar el cursor sobre la letra que se quiera eliminar. Los simples programas de tratamiento de textos le permiten controlar el movimiento del cursor por medio de las teclas con flechas que apuntan en la dirección donde el cursor se moverá. En el Commodore 64, la mayoría de los programas utiliza las dos teclas de cursor que se encuentran en la parte inferior derecha del teclado. Apretada la tecla con flecha, el cursor se moverá en la dirección contraria, usando la tecla de cambio (SHIFT) para controlar el cursor le ahorra al Commodore el coste de añadir otras dos teclas al teclado, pero es torpe para tratamiento de textos, ya que las teclas de cursor se usan muy frecuentemente.

- Movimiento caracter por caracter. El cursor se mueve caracter por caracter.
- Movimiento por caracter - repetición automática. Apretada la tecla el cursor se mueve hasta que se suelte la tecla.
- Movimiento por palabras. El cursor se mueve hasta la primera letra de la próxima palabra.
- Movimiento por frases. El cursor se desplaza hasta la primera letra después del caracter de puntuación final.
- Movimiento por párrafos. El cursor se desplaza hasta el principio del próximo párrafo.
- Principio/final de la línea. El cursor se desplaza al principio o al final de la línea.
- «Home» - posición inicial. El cursor vuelve a su posición inicial, el extremo superior izquierdo de la pantalla.
- Principio. El cursor se posiciona en el principio del documento.
- Fin. El cursor se posiciona al final del documento.
- Salto rápido. El cursor salta adelante o atrás en el documento sobre varias líneas.

Búsqueda y sustitución

Si uno deletrea mal una palabra y esa palabra aparece varias veces en el documento, se le puede indicar al programa que busque y corrija el error? ¿Si uno tiene un documento extenso en la memoria, se puede buscar una sección en concreto indicando una palabra o una frase clave de esa sección? Hay varias variantes al tema de búsqueda y sustitución.

- Buscar palabra clave. Busca una palabra indicada y desplaza el cursor al punto donde la encuentra.
- Sustitución única. Encuentra la palabra clave y la cambia por otra palabra indicada.
- Sustitución selectiva. Encuentra la palabra clave y pregunta si la quiere cambiar por la palabra indicada. Se puede contestar sí o no.
- Sustitución global. Encuentra la aparición múltiple de la palabra clave y las cambia todas por una



Uno de los procesadores recientemente presentado en Estados Unidos

nueva especificada.

- Sustitución global selectiva. Igual como la sustitución selectiva, sólo que repasa todo el documento en búsqueda de la palabra clave.
- Búsqueda libre («Wild card»). Le permite especificar la palabra clave con letras libres. Por ejemplo, indicando que busque la palabra Th#mes, significa que tanto la palabra Thames como Themes serán encontradas ya que la tercera letra es letra libre. Cualquier letra estará aceptada en esta posición.

Características de paginación e impresión

- Los parámetros de página programables. ¿Se puede controlar el número de líneas, longitud de una línea, y el margen superior e inferior en una página?
- Selección de impresora. ¿Funcionará el programa con muchas impresoras distintas o es limitado a sólo una o dos?
- Subrayado. ¿Se puede subrayar el texto?
- Texto resaltado. ¿Se puede resaltar el texto?
- Grandes caracteres. ¿Se pueden introducir caracteres expandidos o más grandes que los caracteres normales en las impresoras que tienen esa opción?
- Itálica. ¿Se puede producir parte del texto en itálica en impresoras con esa opción?
- Margen derecho e izquierdo. ¿Se puede ajustar el texto de manera que el margen derecho e izquierdo sean rectos?
- Subscript/Superscript. ¿Tiene la característica de Suscript/Superscript?
- Cabeceras. ¿Le permite el programa especificar un texto que aparecerá en la parte superior y/o inferior de una página?
- Numeración de páginas. ¿Numera el programa las páginas automáticamente?
- Salto automático de línea. ¿Continuará el programa el texto en la siguiente línea una vez llenado el espacio de la línea actual? ¿Desplaza toda la palabra que se está tecleando o rompe la palabra dejando cada parte en una línea diferente? Romper palabras es un pobre sustituto del salto automático que desplaza la última palabra tecleada en una línea, la que está demasiado larga y no cabe, al principio de la línea.

Algunos procesadores de texto para el 64 tienen todas esas características y unas cuantas más. Pero la relación anterior le da un punto de partida a la hora de evaluar programas potenciales. Hay otras dos características a considerar.

Algunos programas comprueban cada palabra si está bien deletreada o no. Pero no le ayudarán en caso de que usted quiera decir «efectivo» en lugar de «afectivo». Las dos palabras son correctas y serán aceptadas por el programa. Pero palabras como «afectivo» o «efectivo» serán encontradas. Otros programas controlan el uso de las palabras y la gramática y sugieren mejoras. No ha sido posible para nosotros encontrar un programa para el 64, que controla la gramática, pero hemos visto varios que controlan la ortografía.

Autoejecución

Existen juegos o utilitarios que se cargan en la C-64 con el formato de LOAD, 1 es decir LOAD "nombre",8,1.
Esto produce que, además de cargar el programa, se ejecute.
Este pequeño programa realiza lo descripto. Primero se deberá ejecutar normalmente. Luego, si los valores en los DATA son correctos, el programa nos pedirá que borremos las líneas 11, 12 y 13. Luego de esto deberemos hacer RUN. A continuación se debe cargar en memoria el programa elegido y efectuar los POKE que aparecen en las instrucciones. Una vez hecho ésto hay que cargar el programa utilizando el SAVE normal. En este punto la computadora está inoperante; es por ello que debemos resetearla.
Cuando efectuemos LOAD "nombre", 8,1 el programa anterior se cargará y se ejecutará automáticamente.

```
10 REM AUTOEJECUCION
11 FORX=1 TO48: READY:
SUM=SUM+Y: NEXT
12 IFSUM<>5991 THENPRINT
"ERROR EN DATAS": STOP
13 PRINT "BORRA LAS LINEAS
11, 12 Y 13": END
20 FORA=679TO726: READB:
POKEA, B: NEXT
30 PRINT "PARA EFECTUAR
UNA COPIA DE AUTOSTAR EN
UN PROGRAMA"
40 PRINT "CARGALO, BORRA LA
PANTALLA"
50 PRINT "Y LLEVA EL CURSOR
AL FINAL DE LA LINEA"
60 PRINT "LUEGO EFECTUA
ESTE COMANDO"
65 PRINT "POKE43, 166"
70 PRINT "POKE44,2: POKE770,
167: POKE771,2: SAVE NOMBRE
DEL ARCHIVO,8"
80 PRINT "SIEMPRE SE DEBE
CARGAR CON LOAD NOMBRE
DEL ARCHIVO, 8, 1"
90 DATA 169, 147, 32, 210, 255, 169,
82, 32
91 DATA 210, 255, 169, 85, 32, 210,
255, 169
92 DATA 78, 32, 210, 255, 169, 19,
32, 210
93 DATA 255, 169, 131, 141, 2, 3,
169, 164
94 DATA 141, 3, 3, 169, 1, 141, 198, 0
95 DATA 169, 13, 141, 119, 2, 76,
131, 164
```

Eliminador de ? en imput

Si ejecutamos este programa veremos que el signo de interrogación no aparece al lado del mensaje respectivo:

```
10 POKE19, 64: INPUT "INGRESA
TU NOMBRE": ";NOM$: PRINT:
POKE 19,0
20 PRINTNOM$
```

El secreto reside en que, antes de que se ejecute la sentencia INPUT, modificamos el contenido de la dirección que le indica al intérprete Basic si debe ir o no el signo en cuestión (dirección decimal 19). Notemos que luego del INPUT se ejecuta un PRINT y se setea a cero el contenido de la dirección anterior.
De otra forma se efectuaría el retorno de carro pero no se pasaría a la línea siguiente.

Subrutinas

Generalmente cuando desarrollamos algún programa específico en Basic debemos utilizar procedimientos que realicen alguna determinada tarea, la cual es usada por los diversos módulos del programa, si éste lo desarrollamos siguiendo un diseño modular.
Estas tareas pueden ser, por ejemplo, ordenar registros acorde a algún campo de mayor a menor, es decir efectuar lo que generalmente se conoce con el nombre de SORTING. También podemos necesitar una subrutina que formatee los registros a una determinada cantidad de caracteres o, simplemente, que chequee una determinada entrada para que el valor ingresado sea compatible con la variable asignada a tal efecto. En algunas ocasiones estas subrutinas pueden llegarse a utilizar en más de un programa. Es decir que nos sería útil poder almacenarlas en cinta o en diskette para usarlas en el momento requerido. Pero, como ustedes saben, cuando cargamos un determinado programa en la memoria de la máquina, éste destruye al anterior (si lo hubo).
Aquí les ofrecemos un pequeño programa que les permitirá efectuar el MERGE correspondiente. Es decir que si tenemos las subrutinas almacenadas en algún medio externo, sólo debemos escribir el programa principal y luego hacer lo que más abajo se describe para que la subrutina seleccionada pueda acoplarse al final de aquél, evitando de esta forma volver a tipearla. Así podremos tener una especie de librería constituida por las subrutinas más utilizadas por nosotros al escribir programas Basic.
Primero carguen el programa principal, luego agreguen las siguientes líneas al principio de él.

```
1 PRINT PEEK (43), PEEK (44)
2 SV=PEEK (45) 256*PEEK (46)-2
3 POKE 43, SVAND 253: POKE 44,
SV/256: NEW
```

En la línea 1 imprimimos la dirección de inicio del texto Basic. En la C-64 nosotros deberemos ver 1 y 8, al menos que ustedes hayan modificado el inicio del programa por algún motivo. Ahora carguen la subrutina seleccionada y luego efectúen:

```
POKE 43, 1: POKE 44, 8
```

Si los valores impresos en la línea 1 son 1 y 8.
Caso contrario efectúen los POKEs anteriores con los valores nuevos. Por otro lado si corremos varias veces esta rutina podremos agregar más subrutinas. La única precaución que hay que tener es que siempre los números de línea de la subrutina a agregar deben ser mayores que la última línea del módulo anterior, el cual está en memoria.
Para culminar les dejamos un ejemplo para que lo analicen:

```
1 PRINT PEEK (43), PEEK (44)
2 SV=PEEK (45)+256*PEEK (46)-2
3 POKE 43, SVAND253: POKE 44,
SV/256: NEW
10 REM MODULO PRINCIPAL
20 INPUT "INGRESE NUMERO
DE REGISTROS"; NU%
30 IDNU%>1200 THEN 20
40 GOSUB 1000: REM ORDENA
50 END
READY
RUN
1      8
READY
LOAD "ORDENA", 8 (según sea disk
o tape)
(mensajes correspondientes)
READY
POKE 43, 1: POKE 44,8
READY
LIST
1 PRINT
PEEK (43), PEEK (44)
2 SV=PEEK (45)+256*PEEK (46)-2
3 POKE 43, SVAND 253: POKE 44.
SV/256: NEW
10 REM MODULO PRINCIPAL
20 INPUT "INGRESE NUMERO
DE REGISTROS"; NU%
30 IDNU%>1200 THEN 20
40 GOSUB 1000: REM ORDENA
50 END
1000 REM SUBRUTINA ORDENA
1010 (sigue el programa)
1020
1100 RETURN
```

Cuando completen el MERGE no olviden borrar la línea 1,2 y 3 antes de ejecutar el programa.

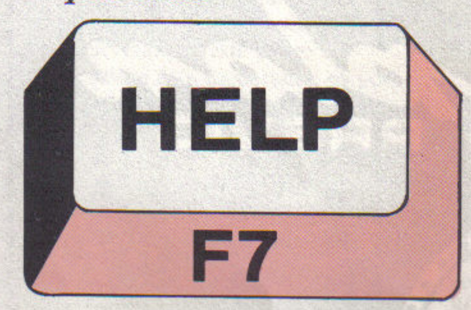
Recuperador de programas

Si, por error, tipeamos el comando NEW durante la carga de un programa, podremos recuperarlo haciendo:

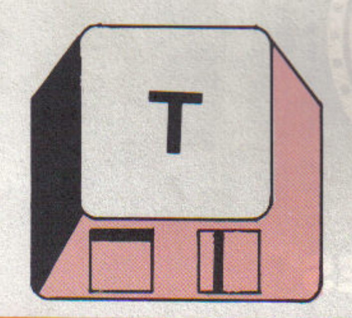
```
POKE2050,1: SYS42291
```

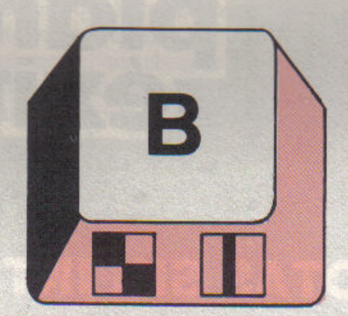
INSTRUCCIONES PROPIAS

Explicamos a fondo los comandos que ofrece esta computadora, empezando por HELP, DIRECTORY, DSAVE, DLOAD, y comentamos cómo definir "ventanas" en la pantalla









Teclado y comandos para el disk drive

En el número anterior publicamos una nota en la cual mencionamos, a grandes rasgos, las características de la Drean Commodore 16.
A partir de aquí, y en números sucesivos, explicaremos más a fondo los comandos que ofrece esta computadora.
Hemos visto que el teclado de la C-16 tenía algunas diferencias con el de la 64. Una de ellas era la disposición de las teclas de control del cursor. Otra, y tal vez la que más llama la atención, es la tecla HELP. Generalmente (por no decir siempre) ocurren errores durante la ejecución del programa. Aquí, la computadora nos imprime la clase del error juntamente con el número de línea donde ocurrió. Si presionamos la tecla HELP se imprimirá automáticamente el texto de esa línea sobre la pantalla. De esta manera podemos rápidamente analizar que fue lo que pasó. Algo importante de resaltar es que la línea se imprime resaltando sus caracteres a través de un "FLASHING".
Entrando en la parte de comandos orientados al uso del DISK DRIVE nos encontramos con DIRECTORY. Este imprime en pantalla el directorio del diskette actual sin destruir los datos en memoria (como ocurre en la C-64 en caso de no disponer del DOS).
Para grabar un programa se utiliza el mando DSAVE cuyo formato es DSAVE "NOMBRE DEL PROGRAMA". Nótese que el ",8" no hace falta ya que la D que tiene este comando al principio indica que se trata de una operación que involucra al DISK DRIVE. Para cargar programas se usa el DLOAD cuyo formato es DLOAD "NOMBRE DEL PROGRAMA". Con el DATASSETTE se usan los mismos sin la D inicial.
Para el caso de DIRECTORY siguen valiendo las mismas características como las del LOAD"$",8. Es decir que podemos cargar, por ejemplo, sólo aquellos programas del tipo SEQ, REL, PRG o DEL o transferir a memoria aquel programa cuyas dos letras iniciales comiencen con AB. Todo esto lo podemos realizar haciendo:

```
DIRECTORY "*=PRG"
DIRECTORY "*=DEL"
DIRECTORY "*=SEQ"
DLOAD "AB*"
```

Definición de "ventanas" en pantalla

Estas "ventanas" nos permiten definir una específica área de la pantalla como nuestra área de trabajo. En realidad toda la pantalla puede considerarse como una gran ventana. La C-16 nos permite modificarla. Para ello se debe hacer:
- Mover el cursos hacia el ángulo izquierdo superior del área buscada.
- Presionar la tecla ESC y luego la tecla 'T' (Top).
- Mover el cursor hacia el ángulo derecho inferior del área buscada.
- Presionar ESC y luego la tecla 'B' (Bottom).

A partir de aquí todas las tareas que hagamos (LIST, PRINT, etc.) se imprimirán en nuestra área de trabajo. Podemos manipular el texto dentro de ella utilizando la tecla ESC. Las funciones de edición de pantalla tales como SCROLLING, INSERT, DEL, cambio de posición de la ventana, etc., pueden ser ejecutadas presionando ESC seguida de la tecla correspondiente. Esa tecla puede ser:
A: insert automático.
B: setea el ángulo derecho inferior de la ventana.
C: cancela insert automático.
D: Borra línea actual.
I: inserta una línea.
J: mueve el curso al comienzo de la línea actual.
K: mueve el cursor al final de la línea actual.
L: activa scrolling.
M: apaga scrolling.
N: retorna al formato normal de pantalla.
O: cancela modo insert, comillas, campo reverso y flash.
P: borra desde el comienzo de la línea actual hasta la posición del cursor en el texto.
Q: borra desde la posición del cursor en el texto hasta el final de la línea.
R: reduce la pantalla
T: setea el ángulo izquierdo superior de la ventana
V: realiza SCROLL hacia arriba.
W: realiza SCROLL hacia abajo
X: cancela la función anterior

SNAKER

Con la Drean Commodore 64 debemos tratar de que el gusano coma sin chocar contra las paredes



Podemos jugar con joystick (port 2) o bien con teclado. Las teclas que debemos mover son: w - arriba - a : derecha - x : abajo d : izquierda

Presentación

Al principio la máquina nos pedirá que esperemos un momento.
Posteriormente aparece la presentación y nos darán la siguiente opción: "i" proporcionará las instrucciones al juego o bien al pulsar otra tecla, éste comenzará.

Descripción del juego:

La imagen del juego es un gusano que debe comer "snaker" tratando de esquivar los obstáculos, sin chocar con las paredes del laberinto y tampoco puede retroceder.

Descripción del programa

Líneas	Detalle
10 - 15	Activar gráficos modo alta resolución - Limpiar área
20 - 45	Definición y carga de la pantalla en alta resolución
50	Carga del sprite
52	Declaración y carga de la matriz SN
55 - 65	Habilitación sprite y almacenamiento actual pantalla
70 - 85	Impresión presentación pantalla
90	Obtener una tecla
92 - 100	Declaración variables y bifurcación en caso de A$= "i"
105 - 155	Colocación en forma random de los obstáculos en pantalla
160	Obtener tecla o joystick
165 - 8000	Loop del juego de acuerdo a las alternativas de ganar o pérdida
8000 - 8060	Instrucciones del juego
8999 - 9020	Rutina efecto de sonido
10000 - 10072	Datos para carga de pantalla y sprite

Descripción de variables

Nombre Variable	Descripción
SC	Valor Score
HI	Valor alto puntaje juego
Z	El valor random de la variable TI
K$-K	contiene el valor del teclado o joystick, respectivamente
X - Y	Manejan fila y columna
A$, C, M	Usos varios
S (100),	SN (20,1), B(6) Vectores que manejan los valores del sprite

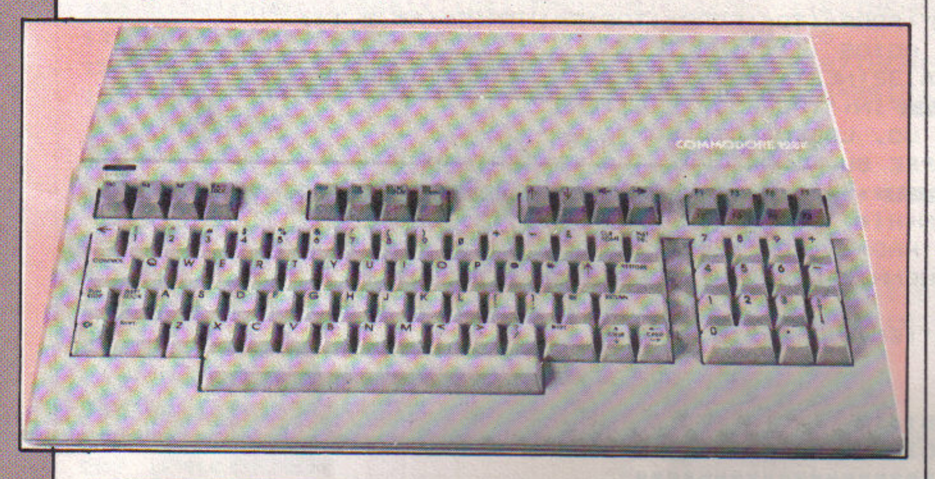
```
10 POKE53280,8:POKE53281,8:PRINT"UN MOMENTO POR FAVOR"
15 POKE52,48:POKE56,48:CLR
20 POKE56334,PEEK(56334)AND254:POKE1,PEEK(1)AND251
25 FORI=0TO511:POKEI+12288,PEEK(I+53248):NEXT
30 FORI=12288TO12295:READA:POKEI,A:NEXT
35 FORI=12504TO12543:READA:POKEI,A:NEXT
40 FORI=12568TO12583:READA:POKEI,A:NEXT
45 POKE1,PEEK(1)OR4:POKE56334,PEEK(56334)OR1
50 FORI=832TO894:READA:POKEI,A:NEXT
52 DIMSN(20,1):FORA=0TO20:READSN(A,0),SN(A,1):NEXT
55 V=53248:POKE53272,(PEEK(53272)AND240)+12:POKE2040,13:POKEV+21,1:POKEV+23,1
60 POKEV+29,1:POKEV+39,5:POKE53280,1:POKE53281,1:PRINT"";
65 POKEV,160:FORA=0TO112:POKEV+1,A:NEXT:A$="        $  SNAKER   $"
70 FORA=1TO11:PRINTSPC(15)MID$(A$,A,10):FORB=1TO50:NEXT:PRINT"";:NEXT
80 PRINT"  PRESIONE I UNA TECLA     "
85 PRINT" OTRA TECLA PARA JUGAR"
90 GETA$:IFA$=""THEN90
```

```
92 DIMS(100),B(6):C=54272:POKE650,255:HI$="DYNAMO  ":TP=10
95 SC=0:HI=100:Z=RND(-TI):POKEV,0:POKEV+1,0:POKEV+21,0:IFA$="I"THENGOSUB8000
100 PL=0:POKE53280,13:POKE53281,13:PRINT"PUNTAJE ";HI
105 FORA=1064TO1103:POKEA,35:POKEA+54272,9:NEXT
110 FORA=1104TO1944STEP40:POKEA,35:POKEA+39,35:POKEA+54272,9:POKEA+54311,9:NEXT
115 FORA=1984TO2023:POKEA,35:POKEA+54272,9:NEXT
120 FORA=1117TO1957STEP40:POKEA,35:POKEA+13,35:POKEA+54272,9
125 POKEA+54285,9:NEXT:FORA=1TO3:POKE1117+40*INT(RND(1)*22),32
130 POKE1130+40*INT(RND(1)*22),32:NEXT:IFTP<40THEN135
131 FORA=1TO20
132 B=INT(RND(1)*880)+1104:IFPEEK(B)<>32ORB=1187THEN132
133 POKEB,35:POKEB+C,9:NEXT
135 FORA=0TO6
140 B=INT(RND(1)*880)+1104:IFPEEK(B)<>32ORB=1187THEN140
145 B(A)=B:POKEB,35:POKEB+C,8:NEXT:FORA=1TOTP+5
150 B=INT(RND(1)*880)+1104:IFPEEK(B)<>32ORB=1187THEN150
155 POKEB,36:POKEB+54272,2:NEXT:S=1:S(0)=1187:H=30:D=1:POKES(0),H:POKES(0)+C,5
157 GOSUB9000
160 GETK$:K=PEEK(56320):IFK$=""ANDK=128THEN185
165 IFK$="W"ORK=126THEND=-40:H=31:GOTO185
170 IFK$="X"ORK=125THEND=40:H=29:GOTO185
175 IFK$="A"ORK=123THEND=-1:H=28:GOTO185
180 IFK$="D"ORK=119THEND=1:H=30
185 IFS<5THENS=S+1:FORA=STO1STEP-1:S(A)=S(A-1):NEXT:S(0)=S(0)+D:GOTO190
187 POKES(S),32:FORA=STO1STEP-1:S(A)=S(A-1):NEXT:S(0)=S(0)+D
190 Z=PEEK(S(0)):IFZ=36THENGOSUB6000
195 POKES(1),27:POKES(1)+C,5:IFZ=35ORZ=27THEN7000
200 POKES(0),H:POKES(0)+C,5:M=0
205 Y=INT(RND(1)*7):IFRND(1)<.5AND(PEEK(B(Y)-1)=32ORPEEK(B(Y)-1)=27)THENM=-1
207 IFM<0THEN230
210 IFRND(1)<.5AND(PEEK(B(Y)+1)=32ORPEEK(B(Y)+1)=27)THENM=1:GOTO230
215 IFRND(1)<.5AND(PEEK(B(Y)+40)=32ORPEEK(B(Y)+40)=27)THENM=40:GOTO230
220 IFRND(1)<.5AND(PEEK(B(Y)-40)=32ORPEEK(B(Y)-40)=27)THENM=-40:GOTO230
225 GOTO160
230 X=0:IFPEEK(B(Y)+M)<>32THENX=1
235 POKEB(Y),32:B(Y)=B(Y)+M:POKEB(Y),35:POKEB(Y)+C,8:IFX=0THEN160
240 GOTO7000
4999 REM
5000 FORA=1TO3:FORB=0TO13:POKE53280,B:NEXTB,A:PRINT"";
5005 FORA=1TO24:PRINT"                                        ";
5010 NEXTA:PRINT"";
5015 FORA=1984TO2023:POKEA,32:NEXT
5020 PRINT"CONGRATULACIONES !!! UD. GANO "
5030 PRINT"SCREEN";TP/10
5032 PRINT"";TP;" PUNTAJE OBTENIDO !"
5035 PRINT" $  LISTO PARA OBTENER > PUNTAJE"
5040 SC=SC+TP:TP=TP+10:FORA=1TO4000:NEXT:PRINT"";:A$=""
5045 FORA=1TO4:FORB=1TO6:PRINTMID$(A$,B,1);
5050 PRINT"          AQUI VAMOS!! ";:NEXTB,A
5055 FORA=1984TO2023:POKEA,32:NEXT:FORA=1TO50:FORB=14TO13STEP-1:POKE53280,B
5060 NEXTB,A:GOTO100
5999 REM
6000 FORA=1TO10:FORB=14TO13STEP-1:POKE53280,B:NEXTB,A:SC=SC+1:PL=PL+1:S=S+1
6005 PRINT"";SC:IFPL=TPTHEN5000
6010 RETURN
6999 REM
7000 FORA=1TO3:FORB=0TO13:POKE53280,B:NEXTB,A:PRINT"";:A$=""
7005 FORA=1TO4:FORB=1TO6:PRINTMID$(A$,B,1);
7010 PRINT"              JUEGO TERMINADO ": NEXTB,A:PRINT"";
7015 FORA=1984TO2023:POKEA,32:NEXT
7020 FORL=CTOC+24:POKEL,0:NEXT:POKEC+5,33:POKEC+12,33:POKEC+19,33:POKEC+6,255
```

```
7025 POKEC+13,255:POKEC+20,255:POKEC+24,15:POKEC+4,33:POKEC+11,33:POKEC+18,33
7030 FORA=0TO2:FORB=8-ATO4-ASTEP-1:POKEC+1,B:POKEC+8,B:POKEC+15,B:POKEC,6
7035 FORW=1TO100:NEXTW,B,A:FORL=CTOC+24:POKEL,0:NEXT
7040 PRINT"██";:FORA=1TO23:PRINT"                              ";
7045 NEXTA:PRINT"████████████████████OH MAL!! "
7050 PRINT"█UD. CHOCO!!!! ";
7055 PRINT"██████████ SU PUNTAJE  ";SC;"  PUNTOS "
7060 IFSC>HITHENGOSUB7100
7062 PRINT"██████ EL PUNTAJE ALTO █";HI;"PUNTOS "
7064 PRINT"██████FUE EL PUNTAJE : ";HI$
7065 PRINT"████████POR FAVOR PRESIONE  █Y PARA JUGAR OTRA VEZ" ;
7070 PRINT"██████████████████O █N █ PARA STOP"
7075 GETA$:IFA$<>"Y"ANDA$<>"N"THEN7075
7080 IFA$="Y"THEN7150
7085 PRINT"█";:A$="██████":FORA=1TO4:FORB=1TO6:PRINTMID$(A$,B,1);
7090 PRINT"              OK ADIOS !!!";:NEXTB,A
7095 FORA=1984TO2023:POKEA,32:NEXT:FORA=1TO4000:NEXT:POKE53280,14
7098 POKE53281,6:PRINT"████FIN:":POKE53272,21:END
7100 PRINT"█ BUENA JUGADA UD. LOGRO ALTO PUNTAJE "
7105 PRINT"█ POR FAVOR TIPEE SU NOMBRE PARA LA POSTERIDAD "
7110 INPUT"██████████████████*************██████████████████";HI$:PRINT
7115 HI$=MID$(HI$,3):IFLEN(HI$)>12THENHI$=LEFT$(HI$,12)
7117 IFLEN(HI$)=0THENHI$="HAGA ALGO ":GOTO7125
7120 IFRIGHT$(HI$,1)="*"THENHI$=LEFT$(HI$,LEN(HI$)-1):GOTO7117
7125 HI=SC:RETURN
7150 PRINT"█";:A$="██████":SC=0:TP=10:GOTO5045
7666 END
7999 REM
8000 POKE53280,12:POKE53281,12
8005 PRINT"██$$$█ INSTRUCCIONES PARA JUGAR █SNAKER█ $$$"
8010 PRINT"█##################################"
8015 PRINT"█ EN ESTE JUEGO UD DEBE MOVER UN SNAKE "
8020 PRINT"█ALREDEDOR DE LA PANTALLA, COLECCIONANDO LOS SNAKE "
8025 PRINT"TRATANDO DE TOMARLOS Y "
8030 PRINT"BORDEANDO█LAS PAREDES Y LOS BLOQUES"
8031 PRINT"█ NO PUEDE RETROCEDER"
8035 PRINT"LAS TECLAS PARA USAR SON :"
8040 PRINT"██ W-A-X-D  O PRESIONE JOYSTICK  EN PUERTO 2"
8055 PRINT"███ POR FAVOR PRESIONE UNA TECLA PARA█ SNAKER█"
8060 GETA$:IFA$=""THEN8060
8065 RETURN
8999 REM***MUSIC AL COMIENZO JUEGO**
9000 FORL=CTOC+24:POKEL,0:NEXT
9005 POKEC+3,8:POKEC+5,41:POKEC+6,89:POKEC+14,117:POKEC+18,16:POKEC+24,143
9010 FORA=0TO20:FR=SN(A,0):DR=SN(A,1)
9015 POKEC+4,65:FORT=1TODR*2:FQ=FR+PEEK(C+27)/2:HF=INT(FD/256):LF=FQAND255
9020 POKEC,LF:POKEC+1,HF:NEXT:POKEC+4,64:NEXT:FORL=CTOC+24:POKEL,0:NEXT:RETURN
10000 REM**DATAS
10015 DATA60,66,153,161,161,153,66,60
10016 DATA126,255,255,255,255,255,255,126
10017 DATA30,63,79,255,255,79,63,30
10018 DATA126,255,255,255,219,90,60,24
10019 DATA120,252,242,255,255,242,252,120
10020 DATA24,60,90,219,255,255,255,126
10021 DATA255,129,169,149,169,149,129,255
10022 DATA0,60,126,114,78,126,60,0
10023 DATA3,255,190,13,85,65,58,170,129,85,85,85,170,170,193,213,255,170,106
10025 DATA128,28,53,112,8,15,254,20,0,0,0,0,0,0,0,0,0,0,0,128,127,248,192,14
10026 DATA172,176,1,86,111,255,170,53,85,86,26,170,172,13,85,88,3,255,224
10070 DATA9634,2,10207,2,9634,2,8583,2,9634,2,5407,4,9634,2,10207,2,9634,2
10071 DATA8583,2,9634,2,5103,4,9634,2,10207,2,9634,2,8583,2,9634,2, 4817,6
10072 DATA10207,6,10207,6,10814,12
```

TRES MAQUINAS EN UNA

Dream proyecta lanzar en el mercado argentino la computadora totalmente compatible con la C64, con 128 K de RAM, un nuevo super Basic y que puede correr programas para CP/M.



Finalmente, y desde hace ya algunos meses, se encuentra disponible en el mercado americano y ya se ven algunas en nuestro país: la nueva Commodore 128. Podríamos afirmar que en una misma consola se encuentran alojadas tres computadoras distintas:
1.- Una Commodore 64 standard, con 64K de RAM y totalmente compatible con la tradicional Commodore 64. (Software y Hardware).
2.- Una versión mejorada de la C64 con 128K de RAM, y un nuevo super-Basic con algunos comandos de alto nivel.
3.- Una computadora basada en el microprocesador Z-80 que puede correr programas escritos para CP/M (Programa de Control para Microcomputadoras) y que permite la utilización de miles de programas disponibles.
Estas "tres" computadoras se encuentran alojadas en una consola plástica color gris, con 92 teclas ordenadas tipo Selectric y que proveen una confortable sensación al ser operadas.
Mirando la máquina por detrás nos encontramos con los mismos conectores de la C-64: Port de expansión, Port de cassette, Port serie, Salida de video, Port de usuario, Port de joystick. Lo único adicional es un conector para monitor color RGB y un muy útil botón de Reset.
Internamente la C-128 tiene 3 diferentes microprocesadores: el tradicional 6510 para operar en modo C-64, 8502 (modo 64 y 128) y un Z-80 para el modo CP/M.
Existen además dos distintos chips de video: el VIC-II para el modo 64 y el modo 128 de 40 columnas y otro chip adicional para el modo 128 de 80 columnas y modo CP/M.
En lo que respecta a la memoria existen bancos separados de ROM y según en qué modo se esté trabajando existirán distintas configuraciones de mapas de memoria.
La nueva Commodore puede operar en modos diferentes:
- Modo 64.
- Modo 128 con 40 columnas de video.
- Modo 128 con 80 columnas de video.
- Modo CP/M.

En el modo 64 la máquina se convierte en una C-64 standard con su conocida leyenda de Basic 2 y 38911 bytes libres. De más está decir, que todo el software disponible para la C-64 es 100% compatible con este modo de operación.
En el modo 128 es cuando realmente se ve la potencia de esta nueva máquina. En este modo se activa el microprocesador 8502 que es compatible con el 6502 y el 6510 pero que corre al doble de velocidad de ambos. Al arrancar en este modo la pantalla nos indicará que tenemos disponibles más de 122 Kbytes libres!, además poseeremos 48 Kbytes de ROM para el intérprete Basic.

En este modo se activa el Basic 7.0, siendo el mismo uno de los intérpretes Basic más poderosos que se puedan encontrar, hasta en máquinas mucho más costosas.
Este Basic posee todos los comandos de la C-64 más los que se encuentran en el Simon's Basic y los que se pueden obtener del Basic 3.5 de la Commodore Plus 4.

El nuevo Basic posee más de 140 comandos que permiten dibujar gráficos, definir e imprimir movimientos a sprites, crear sonidos y música sin tener que realizar los cansadores PEEKs y POKEs ya que este Basic posee 10 sonidos musicales predefinidos.

También se han mejorado los comandos de manejo de disco: DLOAD y DSAVE permiten eliminar el ",8";CATALOG permite leer el directorio sin borrar el Basic; COPY realiza la copia de un archivo cuando se utiliza drives dobles; BACKUP duplica un disco; HEADER formatea un disco.
Otros comandos permiten que un programa se cargue automáticamente sin tener que realizar el fatigoso LOAD"*",8,1 Return RUN REturn !.
A continuación listamos algunos de los nuevos comandos:

AUTO numeración automática de línea.
BACK UP copia de "back up" de un drive a otro.
BANK selecciona bancos de memoria por software.
BOX dibuja una caja en la pantalla.
CATALOG obtiene el directorio de un disco.
CIRCLE dibuja círculos y elipses.
COLLISION detecta colisión de sprites.
DIRECTORY ídem a CATALOG.
DLOAD carga un programa de disco.
DO/LOOP/WHILE/UNTIL/END permiten el control estructurado de lazos.
DSAVE salva un programa en disco.
ENVELOPE define la envolvente de un sonido.
HEADER formatea un disco.

HELP muestra la línea en que ha ocurrido un error.
KEY define o lista las teclas de funciones.
MONITOR entra en el monitor de código de máquina.
MOVSPR ubica y permite el movimiento de sprites.
PRINT llena un área de color.
PLAY define y toca notas musicales.
PRINT USING formatea la salida de impresión.
PUDEF redefine símbolos en PRINT USING.
RENAME da nuevos nombres a archivos.
RENUMBER re-numera las líneas Basic.
SCALE modifica la escala en modo gráfico.
SOUND crea efectos sonoros.
SPRITE define los atributos de sprites.
TEMPO define la duración de notas musicales.
TROFF/TRON permiten trabajar con TRACE.
VOL fija el volumen de salida.
WINDOW define la "ventana" de pantalla.

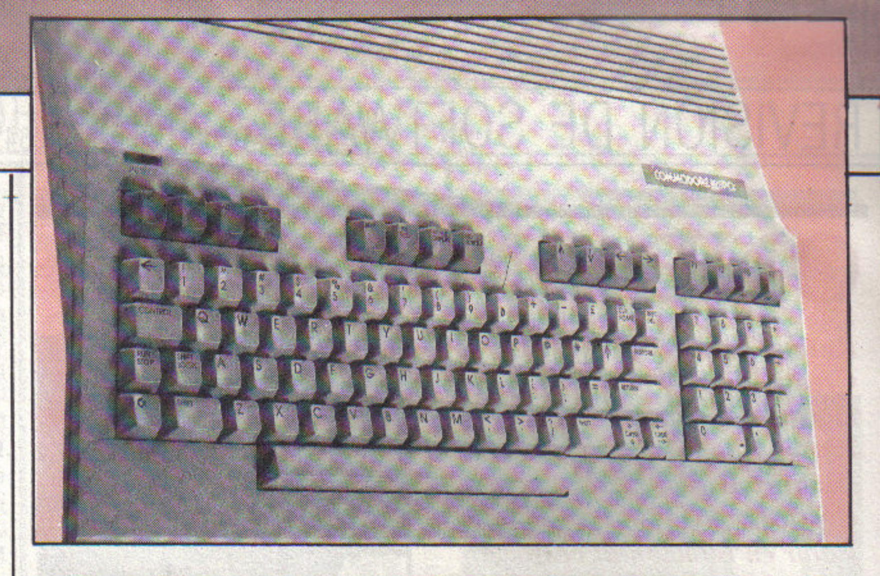
Finalmente digamos que la C-128 posee un monitor de código de máquina residente que permite ensamblar y desensamblar código de máquina, buscar un string, mostrar los contenidos de memoria o de registros, modificar los mismos, guardar programas, etc., todo esto hará feliz al programador en código de máquina.

En el modo CP/M la C-128 puede correr gran cantidad de programas escritos para este sistema como por ejemplo: WordStar, dBaseII y otros muchos más. Los formatos compatibles de disco son: Kaypro, Osborne, IBM VER1, IBM VER2, IBM CP/M 86 y otros más. El modo CP/M puede operar con 40 u 80 columnas y los chips de video y sonido también se pueden acceder desde este modo.

Una de las ventajas de poder tener disponible el modo CP/M es la de contar con software disponible hasta que las empresas proveedoras de software provean los programas correspondientes escritos en modo 128.

Merece un renglón aparte la nueva disquetera disponible para la C-128: el modelo 1571. Si bien la C-128 funciona correctamente con el viejo modelo 1541, es también tan lento como aquel lo era con la C-64. Por otro lado, el nuevo drive con la C-128 funcionando en modo 128 y CP/M donde la velocidad de transferencia es de 5200 cps, o sea 10 veces más rápido que la del 1541.

La disquetera puede trabajar con discos de doble lado/simple densidad y almacena hasta 350 Kbytes de información.



Ficha Técnica de la C-128

CARACTERISTICAS GENERALES:
* Consola de líneas de avanzada.
* 100% compatible con Commodore C-64.
* RAM expandible hasta 512K.
* Doble juego de caracteres.
* Tres modos separados de operación.

MODO 64.
* Microprocesador 6510 (1.02 MHz).
* Chip de sonido 6581.
* 64 K RAM.
* 16 K ROM.
* BASIC 2.0.
* 40 por 25 líneas (resolución de 320 por 200 pixeles).
* 16 colores.
* 8 Sprites.

MODO 128:
* Microprocesador 8502 (1 o 2 MHz, compatible con 6502).
* Chip de sonido 6581.
* 128 K RAM (expandible a 512K utilizando la opción de disco RAM.
* 48K ROM + 16K ROM para soporte DOS.
* BASIC 7.0.
* Monitor de código de máquina.
* 40 por 25 líneas (resolución de 320 por 200 pixeles).
* 80 por 25 líneas (resolución de 640 por 200 pixeles).
* 16 colores + 8 sprites (sólo 40 columnas).

MODO CP/M:
* Microprocesador Z-80 (2.04 MHz).
* CP/M Plus Versión 3.0.
* 128K RAM (expandible a 512K utilizando la opción de disco RAM.
* 40 por 25 líneas (320 por 200).
* 80 por 25 líneas (640 por 200).
* 16 colores.

TECLADO:
* Teclado tipo QWERTY de 92 teclas.
* 14 teclas numéricas.
* 8 teclas de funciones programables.
* 6 teclas de cursor.
* Tecla HELP.
* Tecla 40/80 columnas.
* No Scroll.
* Escape.
* Mayúscula.

ENTRADAS/SALIDAS:
* Port de usuario.
* Port de cassette.
* Port de RF/TV.
* Entrada de audio.
* Salida de video compuesto.
* Port serie.
* 2 Ports de juegos.
* Port de Cartridge.
* Salida de audio.
* Salida p/monitor RGBI.

PERIFERICOS RECOMENDADOS:
* Impresoras MPS 802, MPS 803.
* Drive de disco 1571.
* Monitor monocromo 1901.
* Monitor color digital RGBI 1902.
* Modems 1660 y 1670.
* Totalmente compatible con accesorios para C-64.

DIMENSIONES:
* 56 mm por 432 mm por 324 mm.

PESO:
* 11.23 Kg.

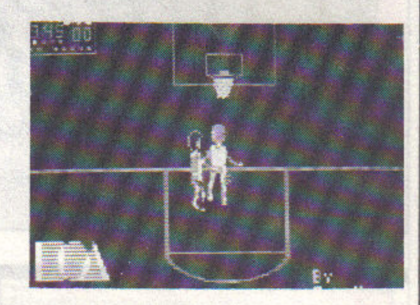
COLOR:
* Beige claro.

ALIMENTACION:
* 117 volts CA, 60 Hz, 15 watts.

G.E.

ONE ON ONE

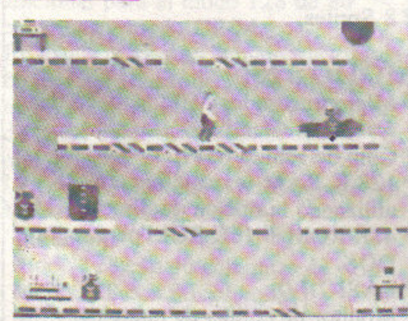
Rating Total: A
Creatividad: A⁻
Documentación: B⁺
Profundidad del juego: B
Desafío: Difícil
Gráficos: A
Valor en relación al costo: A
Mantiene el interés: A



Este entretenido juego deportivo simula ser un partido de basquet en una cancha pequeña para dos jugadores. Los dos simpáticos personajes, Julius (Dr. J) Irving y Larry Bird aportan su sabiduría, habilidad y estilo.
Pueden jugar uno o dos participantes. Si son dos, lo harán entre sí; de lo contrario la computadora tomará el lugar del segundo participante.
Continuamente se verá cómo Larry pone en práctica su imparable salto para poder embocar en el cesto, mientras que Dr. J realiza increíbles demostraciones de destreza y ocasionalmente logrará un tanto después de un sorprendente giro de 360°. Larry, más alto y robusto que Dr. J, bloqueará muchos de los envíos y los empujará fuera del área del cesto.
La presencia de un referee agrega realismo al juego sancionando fouls cuando los jugadores hacen picar la pelota o interfieren intencionalmente a su oponente. Cada jugador salta y bloquea los envíos girando mientras puede avanzar y arrojar la pelota al cesto. Como en un partido real, el jugador tiene un tiempo límite para efectuar el tiro y que en este caso no puede exceder los 24 segundos. El que se detiene demasiado en un juego de piernas se fatigará inútilmente. En cambio si uno irrumpe en el aro del oponente con un disparo sorpresivo probablemente lo vencerá. Las jugadas más sobresalientes podrán volverse a ver en un "replay" instantáneo.
Puede utilizarse en varios niveles, desde aficionado hasta profesional. En el más alto, el jugador de la computadora marcará de muy cerca y a menudo quitará la pelota. A los que recién se inician en el juego les conviene empezar con alguno de los dos primeros niveles y tomar el rol de Larry Bird.
El accionar de palanca es peculiar. Una suave presión en el botón mueve al jugador y una presión más firme y duradera efectuará el tiro.
Mientras muchos se reconocen incapaces o muestran defectos para jugar al basquet, aquí estarán satisfechos, porque en ONE ON ONE pueden exhibir una habilidad que no poseen en la realidad. Eso no quiere decir que sea muy sencillo ganarle a la computadora, pero uno puede conformarse con hacerle un buen partido. La diferencia más importante con el juego verdadero es que aquí, para llevarse la pelota, la habilidad no es muy importante, y el disparo de larga o de muy corta distancia es semiautomático y basado en porcentajes. Se puede realizar un giro de 360° a mitad de camino, tirar la pelota y encestar, o simplemente errarle. Bloquear los envíos del oponente es una tarea muy importante.
ONE ON ONE seguramente será todo un éxito porque simula el juego verdadero con excelentes gráficos realizados por expertos.

MISION IMPOSIBLE

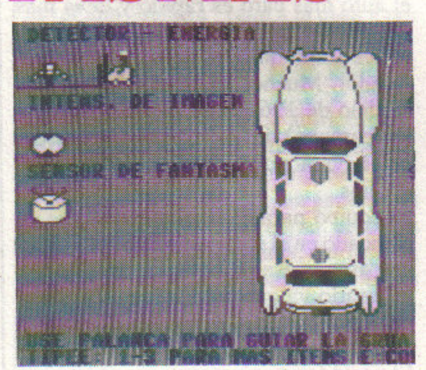
Rating Total: A⁻
Creatividad: B
Documentación: B
Profundidad del juego: B⁺
Desafío: Difícil
Gráficos: A
Valor en relación al costo: B⁺
Mantiene el interés: B⁺



En Misión Imposible, el jugador, como el Agente Especial 4125, deberá penetrar la fortaleza clandestina de Elvin Atombender y detenerlo para que no haga estallar al mundo. Esta no es una misión simple. Para vencer se deberá evitar la guardia de robots fieles al científico loco, encontrar las claves para romper su código de seguridad, encontrar la sala secreta de control y desactivar la computadora "del juicio final" antes de que el tiempo se termine. Si esto parece una sencilla aventura, primero convendrá informarse.
El complejo laberinto subterráneo tiene diferentes niveles conectados por ascensores que llevan a las habitaciones y salas de computación del profesor. Cada uno de los 32 cuartos está particularmente construido, por lo que sólo Elvin puede disponer y manejarse fácilmente en ellos. Todo el mobiliario está suspendido sobre corredores o pasillos, o ubicado sobre pisos que terminan abruptamente cayendo en el vacío. Un paso equivocado y el jugador puede encontrarse cayendo de cabeza en un abismo. Los robots de Elvin patrullan los pasillos constantemente. Entonces hay que encontrar las contraseñas que el distraído Elvin escondió en algún mueble. Algunas desactivan los robots, otras detienen los elevadores desde alguno de los salones. La clave del Salón de Control es mucho más enrredada. Está dividida en pequeñas partes y se deberá usar la computadora de bolsillo, que posee el Agente Especial, para tener éxito. La solución que se obtenga revelará una de las nueve letras que se necesitan para poder ingresar al Salón de Control.
Para triunfar en este juego se requiere una considerable cuota de destreza en el manejo del joystick. Habrá que saltar

sobre el vacío y dar volteretas sobre los robots exactamente en tiempo. Si bien se tiene un innumerable número de vidas, cada pérdida costará 10 minutos de juego y se tendrán sólo seis horas para detener a Elvin. El Agente podrá llamar telefónicamente a la computadora principal de la Central de Inteligencia para solicitar ayuda. La clave de esa computadora le costará dos minutos del precioso tiempo cada vez que la utilice.
El escenario de Misión Imposible es excelente aunque dificultoso. No presenta niveles fáciles para principiantes y difícilmente se llega a vencer. Los gráficos y sonidos son sobresalientes. El juego tiene algunos de los mejores parlamentos simulados que se puedan conocer y llegan a sonar como en los films de terror de Vincent Price. Al caminar por los corredores, los pasos producen un sonido metálico, y los gritos en el momento de caer en los abismos, son realmente escalofriantes. Misión Imposible atrapa el interés e invita a volver a jugar. Por suerte, el alto nivel de dificultad no frustrará al jugador común ni echará a perder su diversión.

LOS CAZAFANTASMAS



Rating Total: B
Creatividad: B
Documentación: C
Profundidad del juego: B
Desafío: Mediano
Gráficos: B
Valor en relación al costo: B
Mantiene el interés: B+

Los Cazafantasmas, una de las más grandes películas taquilleras se convirtió en un video juego que está destinado a ser un hit. Este entretenido programa retiene el espíritu del film combinándolo con una buena música y divertidos efectos de sonido y parlamentos.
El objeto del juego es montar la propia empresa de los Cazafantasmas en el vecindario del jugador. El banco presta diez mil dólares para comprar un

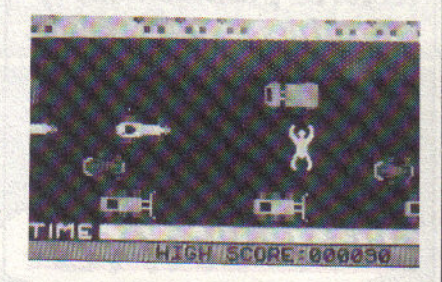
vehículo y equiparlo con importantes elementos para cazar fantasmas. Entre ellos: un detector de energía PK, sensores, un sistema de confinamiento de láser portátil, aspiradora de fantasmas, y trampas y carnadas para atraparlos.
el auto más caro es rápido y lleva más equipamiento, pero generalmente es inalcanzable para el presupuesto del jugador. Muchos se deciden por un modelo de 1963 o un Beetle (Cucaracha) de Volkswagen, a no ser que haya acumulado dinero de juegos previos.
El equipo para cazar fantasmas es colocado en el coche con una grúa y hay que asegurarse de haber comprado lo necesario ya que no puede descargarse nada una vez que se ha finalizado la carga.
El vecindario se presenta en una trama de calles con varios edificios representados por el cuartel de policía o el Templo de Zool. Los fantasmas se deslizan despaciosamente hacia el Templo y algunos se detienen en el camino para merodear por otros edificios. El indicador de energía PK marcará un ascenso. Cuando en un edificio se encienden señales rojas deberá dirigirse el vehículo hacia allí lo más rápido posible. La pantalla cambiará a una visión aérea del camino mientras el auto se acerca al lugar.
El jugador podrá moverse de un lado al otro del camino "aspirando" fantasmas que se cruzan mientras se avanza. Eso siempre y cuando se haya comprado una aspira fantasmas. Para esto se requiere cierta habilidad.
La visión y el efecto sonoro que se producen cuando los fantasmas son aspirados son fantásticos.
Luego se llega al sitio del disturbio para buscar a los fantasmas que entran y salen por las ventanas superiores. Se deberá enviar al primer cazafantasmas del equipo a poner trampas en el centro del edificio. Luego se lo desplazará hacia el costado izquierdo de la pantalla para ponerlo mirando en dirección a la trampa. Allí deberá esperar mientras el jugador coloca al segundo miembro del equipo en el extremo derecho. Cuando el fantasma esté al alcance se deberá accionar el ionizador negativo de los equipos de los hombre, dirigiendo lentamente el flujo hacia adentro. Nunca se deben hacer disparos cruzados. Así se podrá atrapar a la silueta directamente debajo de la trampa.
Una vez que el fantasma está debajo de la trampa, se deberá "disparar" y el fantasma será absorbido por ella. Si se hace bien el cazafantasma gritará "Ghostbusters" y se ganará 300 dólares de honorarios. Pero si pierde uno de los cazafantasmas, se convertirá en una silueta y el nivel de energía PK de la ciudad se incrementará en 300. El hombrecito derrotado gritará en un lamento **¡me transformo en silueta!**
La escena es como en la película donde los cazadores tienen orden de no cruzar los flujos de ionizadores. En esta parte del juego no se necesita mucha habilidad. Los rayos laser están colocados hacia adentro en un ángulo, pero sólo pueden ser usados hacia adentro, donde si se tiene suerte se puede encontrar a un fantasma (si es que aun se encuentra al alcance del rayo).
De todos modos los fantasmas son tontos y volverán a ponerse en el radio de alcance, siempre y cuando no se haya disparado el rayo primero. En cuyo caso el cazador será destruído como se explicó anteriormente.
Capturar un fantasma es una cuestión de precisión, se debe lanzar el rayo en el momento exacto en el que el fantasma esté bajo la trampa. Incluso pueden ser capturados aunque los rayos no se hallen en la posición exacta. Cuando todas las trampas están llenas se regresa al cuartel central.
Cuando el nivel de energía PK está excesivamente alto una señal luminosa. la **"Alerta Marshmallow"** se encenderá y apagará en la parte inferior de la pantalla. Los fantasmas errantes podrán unirse y formar un Marshmallow (un dulce típico de Estados Unidos) gigante que provocará destrozos en la ciudad valuados en cuatro o cinco millones de dólares, los que serán debitados de la cuenta del jugador. A veces se puede detener al Marshmallow utilizando carnada (la tecla B).

El guardián y señor de las llaves comienza a vagar por las calles cuando el nivel de energía llega a cerca de las diez mil unidades. Parecerá que nada se puede hacer al respecto más que culpar al destino. Sin embargo las instrucciones insinuarán que el jugador puede colocar dos de sus hombres en el edificio y hacerlos llegar hasta la parte más alta del Templo. Muchos han intentado ese objetivo pero aparentemente fue en vano, porque nunca sucede nada. El primer cazafantasma levanta la trampa y se retira porque el edificio del templo no está marcado por la señal luminosa que indica la presencia de fantasmas. El juego generalmente termina en una forma frustrante que parece incontrolable. El guardián y el señor de las llaves unen sus fuerzas y como siempre el jugador no tiene más dinero que el que tenía al principio, siendo un prerrequisito para ganar el juego. Tarde o temprano se puede detener el gran Marshmallow y ganar la justa recompensa.

Los Cazafantasmas es un juego que llamará la atención de los fanáticos de las películas entretenidas. Su música, los efectos sonoros y las simpáticas escenas de caza compensan el nivel de reiteración del juego. Los Cazafantasmas posee la cuota de fantasía que los jugadores exigen.

FROGGER

Rating Total: A–
Creatividad: B
Documentación: C
Profundidad del juego: B–
Desafío: Mediano
Gráficos: B+
Valor en relación al costo: A–
Mantiene el interés: B+



Frogger de Commodore 64 es una cercana adaptación del juego del mismo nombre que pertenece a la línea de juegos Sega. El objetivo es llevar una rana en un tiempo limitado a través de cuatro carriles de tránsito, hacerla saltar, luego, sobre las espaldas de tortugas acuáticas, y sobre troncos que flotan en el agua, hasta que llegue segura a casa. La rana puede saltar en carriles alternativos, y también las tortugas y los troncos se desplazan en diferentes sentidos y velocidades.

Cuando las cinco ranas han llegado a casa el juego entra en un nivel más difícil. Después de cinco niveles, el jugador es recompensado con una rana extra.

En los niveles avanzados, el jugador es amenazado con el paso de autos que se trasladan por espacios estrechos en los carriles rápidos, con mortíferas serpientes que se arrastran por la banquina y los troncos, y por hambrientos cocodrilos que se mueven en la corriente del río. Los troncos se deslizan más rápido y la cantidad aumenta, en tanto las tortugas desaparecen cuando más se las necesita. Hay premios que recompensan al jugador que consigue llevar una "señora" rana a la casa y por engullir insectos. El juego tiene una velocidad rápida y otra lenta para que todos puedan jugar.

El sonido y los coloridos gráficos de Frogger hacen de él un juego divertido. Los personajes aparecen a través de todo el juego. La música, en general, es idéntica a la del juego original, pero han sido agregados varios tonos, entonces la música no es muy repetitiva. De todas maneras los jingles se hacen pesados después de un largo rato de estar jugando. Afortunadamente se puede anular la música sin necesidad de eliminar los efectos de sonido.

Fogger es un juego que provoca adicción y diversión, especialmente entre los más pequeños. Se convierte en algo repetitivo después de un rato, pero entretiene y mantiene el interés de la mayoría de los jugadores por un período moderado.

TRUCOS

Teclas de función

Como saben las teclas de función (F1, F2, F3, ... F8) pueden utilizarse para realizar una determinada acción cuando se solicita una entrada por teclado. Por ejemplo, podemos determinar que cuando F1 sea presionada haremos una búsqueda en un archivo dado o si la tecla presionada fue F2 efectuaremos una conversión numérica. Para poder efectuar esto, debemos saber los códigos ASCII de dichas teclas. Ellos son:

F1	133	F2	137
F3	134	F4	138
F5	135	F6	138
F7	136	F8	138

Para visualizar mejor esto, les dejamos un pequeño ejemplo:

```
200 PRINT "OPRIMA F1 PARA
JUGAR DE NUEVO"
210 PRINT "OPRIMA F2 PARA FI-
NALIZAR"
220 GETA$: IFA$="" THEN 220
230 IFA$=CHR$ (133) THEN 10
240 IFA$=CHR$ (137) THENEND
```

Chequeo de entrada

En ocasiones debemos comprobar durante la ejecución de un programa, que cuando solicitamos algún dato desde el exterior (aquél que es ingresado por el usuario) éste sea compatible con la variable respectiva. De otra forma se imprimirían los correspondientes mensajes de error y, en algunos casos, se pararía la ejecución del citado. Aquí les ofrecemos un programa que verifica que el número ingresado sea un número natural (recuerden que éstos son el 1,2,3...)

```
10 INPUT "INGRESE NUMERO
DE OPCION"; OP
20 IFINT (ABS (OP))() OP THEN 10
30 continúa el programa
```

DEF FN

La C-64 almacena las direcciones de memoria en el formato Byte bajo-Byte alto. Si deseamos saber una determinada dirección, por ejemplo la dirección de inicio del área de almacenamiento de variables, debemos hacer:

```
10 DIR=PEEK (45)+256*PEEK (46)
```

De esta manera cada vez que queramos saber una dirección tendremos que cambiar los valores de los PEEKs. Esto podemos evitarlo si usamos la sentencia de definición de funciones (DEF FN):

```
10 DEF FN ADR (A)=PEEK (A)+
256*PEEK (A+1)
20 DIR= FN ADR (X)
```

Donde X es la dirección deseada.

Protección contra escritura

Si ejecutamos:

```
POKE 818,32
```

desactivaremos el comando SAVE (no podrá ser ejecutado). Lo podemos volver a activar haciendo:

```
POKE 818,237
```

Directorio inofensivo

Cuando se usa el comando LOAD "$", 8 el directorio del diskette actual se almacena en la memoria de la C-64 destruyendo, si los hay, datos anteriores. Esto lo podemos evitar tipeando en modo directo:

```
POKE44, PEEK (46) + 1
LOAD "$",8
LIST (se lista el directorio)
```

Luego, cuando deseamos volver a nuestro programa, hacemos:

```
POKE44,8
LIST (se lista nuestro programa)
```

Back Up

- ¿Qué es un back-up?

Jorge Márquez
Capital Federal

Se denomina back-up al método a través del cual el contenido de un soporte (ej. cinta, diskette) se transfiere a otro soporte a fin de tener una copia, evitando inconvenientes en caso de fallas del original.

Formateo de un diskette

- ¿Qué es el formateo de un diskette y cómo se hace?

Mario Suarez
Rosario

Tipear
OPEN 15,8,15,"N0: nombre,zz"
El formateo cumple con la función de asignar al diskette virgen las pistas y sectores disponibles y la habilitación de un directorio.

¿Programa o archivo?

- ¿Qué diferencia existe entre programa y un archivo?

Liliana Tejada
Quilmes

Todo archivo es aquel que almacena datos de utilidad al usuario, tales como: nombres, direcciones, teléfonos, mercaderías, proveedores, etc.
En cambio el programa es el medio que permite al ser ejecutadas las instrucciones u órdenes que lo componen, manejar archivos, efectuar cálculos.
Si lo definimos en métodos más sencillo el programa es la lista de órdenes dadas a la computadora para realizar una acción.

Instrucción equivocada

- ¿Alguna instrucción indebidamente dada al equipo, lo puede dañar?

Ramiro Cifuentes
Santa Fé

La respuesta es llanamente no. El hardware no se daña por una orden a nivel de software dada equivocadamente. A veces sólo puede ocurrir que el equipo se "cuelgue", es decir que de un loop indefinido y en dicho caso se deberá interrumpir la ejecución.

Continuamos con esta sección para que los lectores planteen sus consultas y sugerencias. Para eso deberán escribir a nuestra redacción: Cerrito 1320, 1º piso Buenos Aires (1010).

Archivo relativo

¿Pueden decirme cuál es la forma correcta de abrir un archivo relativo en la C-64?

Carlos Maio
Capital Federal

Para abrir un archivo relativo en la C-64 se debe hacer:
OPEN 3, 8, 3, "NOM,L, + CHR$ (LON)
donde NOM: NOMbre del archivo
LON: LONgitud máxima de los registros

Se deben tener siempre presente las siguientes aclaraciones:
1º) La longitud del nombre no debe superar los 16 caracteres
2º) La longitud de los registros, es decir el parámetro LON, debe ser siempre un número divisible por 256. Lo que hace LON es decirle al DRIVE en cuantas partes debe dividir un Block.
Cada uno de esos pedazos serán los registros de nuestro archivo. Como cada Block está formado por 256 Bytes, el D.O.S. hace una pequeña cuenta de dividir entre éste y la longitud del registro. Por ejemplo si la longitud es de 64 caracteres, cada Block se dividirá en 4 registros, si fuese de 128 caracteres cada Block tendrá 2 registros, etc. Pero si la longitud fuese de 80 el D.O.S. no puede dividir en 3.2 partes o si fuese de 63 no lo puede dividir en 4.063492 partes iguales!.
En estos casos la apertura del archivo no se lleva a cabo y aparecerá una condición de error en el Drive. Al leer éste nos encontraremos con que es "NO CHANNEL".

Basic

¿A qué se debe que un Basic compilable es más veloz que un Basic intérprete standard?

Andrés Ivern
Rosario

El Basic intérprete es mucho más lento que el compilable ya que por cada línea que toma del texto Basic efectúa la correspondiente traducción a código de máquina y la ejecución respectiva. Es decir que siempre traduce a código de máquina por más que esa línea haya sido traducida. Evidentemente esta traducción demanda un determinado tiempo por más pequeño que éste sea.
A grandes rasgos, la tarea de este Basic puede resumirse de la siguiente manera:
1) Tomar siguiente línea
2) Traducir a código de máquina
3) Optimizar el código - compactarlo
4) Ejecutar código de máquina final
Este trabajo se realiza a lo largo de todo el programa.
El Basic compilable (y en general todos aquellos lenguajes compilables) traduce el texto Basic una sola vez. Es decir, pasa el lenguaje fuente (nuestro Basic) al lenguaje objeto (código de máquina) y lo prepara para la ejecución que se realizará sobre este último. De esta manera cada vez que ejecutemos el programa no se realizarán traducciones ya que, como antes dijimos, ésta se lleva a cabo sobre el objeto, sobre el código de máquina. Así, el tiempo de demora por la traducción de cada línea desaparece. Podríamos resumir la función del compilable de la siguiente manera.
1) Traducir todo en texto Basic (programa fuente) en su equivalente en lenguaje de máquina (generar el programa objeto)
2) Compactarlo
La desventaja que tiene este Basic es que necesitamos memoria adicional para almacenar el programa objeto. Esto fue uno de los parámetros más importantes que se considero en el momento de elegir el lenguaje para las Homes Computers. Es por ello que hoy existen en casi todas el Basic intérprete.

Editar

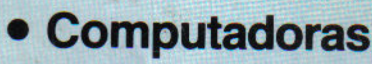
Poseo un Procesador de palabra y quisiera saber cómo puedo hacer para editar un programa Basic.

Francisco Aller
La Plata

Como saben todo procesador de palabra trabaja con archivos secuenciales. Es decir que el texto que se crea es almacenado como un archivo secuencial. Por ello necesitamos cargar el programa deseado en memoria y tipear:
OPEN, 8, 8, "0:Nombre del archivo, S,W": CMD8: LIST
PRINT#8
CLOSE8
De esta forma grabaremos el programa Basic como un archivo secuencial.

A SU DREAN COMMODORE EQUIPELA CON LA LINEA DE ACCESORIOS MAS COMPLETA DE PLAZA

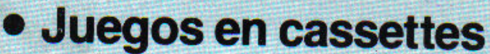
Ud. puede adquirir la línea más completa
de accesorios, programas y periféricos,
en las mejores condiciones y con nuestro exclusivo
plan de financiación de 3 a 12 cuotas.



- **Computadoras**



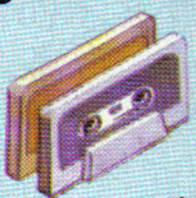
- **Interface para impresoras**
- **Joysticks**
- **Impresoras**



- **Juegos en cassettes**



- **Accesorios**



Lo esperamos porque jamás
la computación estuvo tan cerca suyo.

PLANES ESPECIALES PARA COMERCIOS
TARJETAS DE CREDITO



SYSTEMS

Paraná 223, (1017) Capital Federal
45-6727 / 40-3625.

VICONEX
SU ALIADO EN COMPUTACION

Avda. de Mayo 767, (1084) Capital Federal.
30-3301 / 33-2106 / 34-8371

Sábados abierto medio día